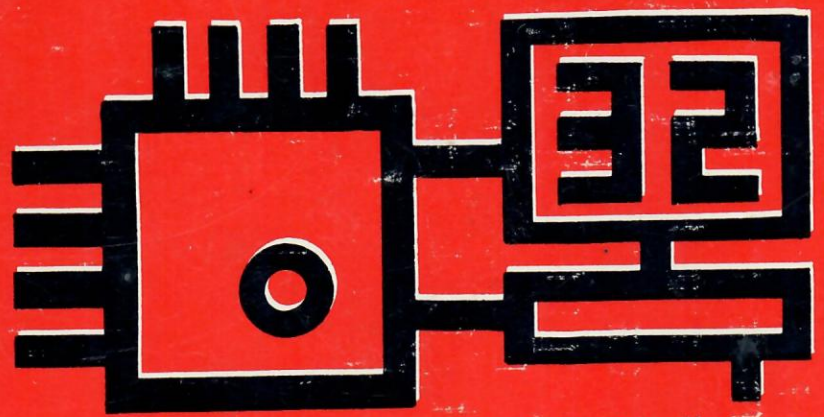
Ed-Lab

EXPERIMENT MANUAL

ADVANCED DIGITAL SYSTEMS

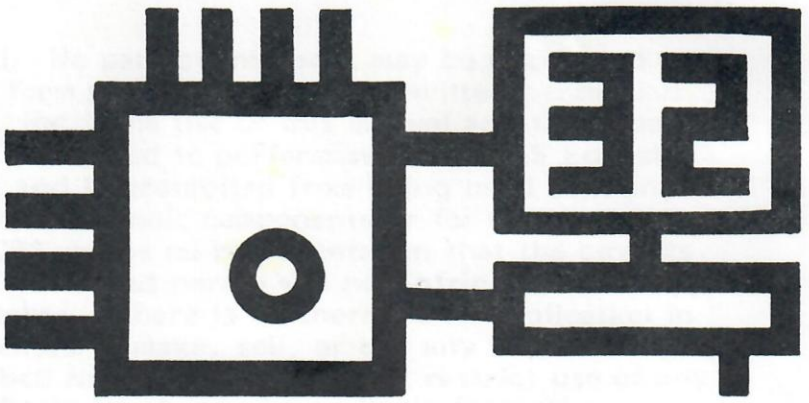


CES INDUSTRIES, INC.

Ed-Lab

EXPERIMENT MANUAL

ADVANCED DIGITAL SYSTEMS



CES INDUSTRIES, INC.

Name _____

Class _____

School _____

Ed-Lab

EXPERIMENT MANUAL

ADVANCED DIGITAL SYSTEMS

Copyright © 1972, 1974, 1977, 1979, 1981 CES INDUSTRIES, INC.

All rights reserved. No part of this book may be duplicated or reproduced in any form without the specific written permission of CES Industries, Inc. The use of this manual and the experiments described is restricted to performance with CES Ed-Lab training equipment and is prohibited from being used with any other training aid or electronic components or for other educational purposes. CES makes no representation that the circuits, logic, or programs described herein will not infringe on existing or future patent rights. There is furthermore no implication in the granting of licenses to make, sell, or use any circuits, logic, or programs described herein. This does not restrict use of any materials or data sheets which may be available from other sources.

ISBN 0-86711-008-2
Printed in United States of America

CES INDUSTRIES, INC.
Educational Products Division
130 Central Avenue
Farmingdale, New York 11735
(516) 293-1420

PREFACE

In the relatively short time since the development of the Ed-Lab 700 Computer System Lab, the advancement in electronic components and systems has continued at lightening speed. The introduction of new technologies and components such as optoelectronics and wider selections of MOS Large Scale Integrated Circuits and Microprocessors have expanded the economical applications of these devices to digital communications, processing and control systems.

The Ed-Lab 900 Memory and Display Lab, uses these modern devices to provide a state-of-the-art training system with Memory, Display, Keyboard and all the LSI and MSI devices necessary to build a complete programmable automatic digital computer and a digital communications system as well as a full Microcomputer system, when used with the Ed-Lab 700 and the Model CES 380 Microprocessor.

The experiments cover all aspects of computer technology including Input, Output, Memory, Display, Programming, Digital Data Communications, Microprocessors and Computer Wiring and Troubleshooting Skills. Only the Ed-Lab 700 serves as a prerequisite and all descriptive material is included in this Experiment Manual which is compatable with many reference texts.

This series is designed to allow the student to really learn all about computers, digital and analog systems by patching together the functional elements. Whether your interest is in logic, computers, programming, building projects, digital communications, digital data systems, digital tape recording, electro-mechanical control or even digital electronic music, the Ed-Lab 900 and the Ed-Lab 700 provide in-depth experimentation opportunities, and make it an interesting, enjoyable, and motivating experience. The addition of the Microprocessor Module expands the system to include microcomputers. This computer technology training system including hardware and software is designed to provide the skills and knowledge that are needed by industry so that the students will obtain rewarding careers.

Norman Nesenoff
President
CES INDUSTRIES, INC.
Educational Division
Farmingdale, New York 11735

SAFETY

Safety is good common sense. The Ed-Lab equipment is designed with all current limited and voltage limited Power Supplies so that there is no danger to the student when touching any wires and no danger of burning-out any of the electronic components.

DO NOT USE AN EXTERNAL POWER SUPPLY WITH THE ED-LAB.

While working in the laboratory, the student is exposed to many different types of equipment, power tools, soldering irons, test equipment, drills, rotating machinery, and so on. The 120 volt power socket can cause a shock, and the shock can result in your jumping or falling. Care should be exercised while working on any equipment including automobiles, television sets, carpentry tools, as well as electricity and electronics.

Do not get involved in any "horsing-around" or in practical jokes. Don't throw anything in the lab. Don't drop the equipment off the bench or on your foot. Don't burn the line cord with a soldering iron. If you remove the Ed-Lab front panel don't touch the 120 volt line-cord connections in the rear while the plug is still connected to the socket.

If an accident occurs, notify your instructor, shut off all power, and administer first aid as necessary. Your instructor should review with you the safetv requirements in your laboratory.

Ed-Lab

FIRST AID for ELECTRICAL SHOCK

If you observe an accident involving electrical shock,
DON'T JUST STAND THERE - DO SOMETHING!

RESCUE OF SHOCK VICTIM

The victim of electrical shock is dependent upon you to give him prompt first aid. Observe these precautions:

1. Shut off the high voltage.
2. If the high voltage cannot be turned off without delay, free the victim from the live conductor. REMEMBER:
   a. Protect yourself with dry insulating material.
   b. Use a dry board, your belt, dry clothing, or other non-conducting material to free the victim. When possible PUSH - DO NOT PULL the victim free of the high voltage source.
   c. DO NOT touch the victim with your bare hands until the high voltage circuit is broken.

FIRST AID

The two most likely results of electrical shock are: bodily injury from falling, and cessation of breathing. While doctors and pulmotors are being sent for, DO THESE THINGS:

1. Control bleeding by use of pressure or a tourniquet.
2. Begin IMMEDIATELY to use artificial respiration if the victim is not breathing or is breathing poorly:
   a. Turn the victim on his back.
   b. Clean the mouth, nose, and throat. (If they appear clean, start artificial respiration immediately. If foreign matter is present, wipe it away quickly with a cloth or your fingers).

   c. Place the victim's head in the "sword-swallowing" position. (Place the head as far back as possible so that the front of the neck is stretched).
   d. Hold the lower jaw up. (Insert your thumb between the victim's teeth at the midline - pull the lower jaw forcefully outward so that the lower teeth are further forward than the upper teeth. Hold the jaw in this position as long as the victim is unconscious).
   e. Close the victim's nose. (Compress the nose between your thumb and forefinger).
   f. Blow air into the victim's lungs. (Take a deep breath and cover the victim's open mouth with your open mouth, making the contact air-tight. Blow until the chest rises. If the chest does not rise when you blow, improve the position of the victim's air passageway, and blow more forcefully. Blow <u>forcefully</u> into <u>adults</u>, and <u>gently</u> into <u>children</u>.
   g. Let air out of the victim's lungs. (After the chest rises, quickly separate lip contact with the victim allowing him to exhale).
   h. Repeat steps f. and g. at the rate of 12 to 20 times per minute. Continue rhythmically without interruption until the victim starts breathing or is pronounced dead. (A smooth rhythm is desirable, but split-second timing is not essential).

DON'T JUST STAND THERE - DO SOMETHING!

CONTENTS

1. INTRODUCTION & COUNTERS

Introduction
1-1. Bit Counter
1-2. Pre-Set Counter
1-3. Cascaded Counter
Trainer Summary

2. DISPLAYS and REGISTERS

2-1. Seven Segment Display
2-2. Binary Coded Decimal Display
2-3. Shift Register
2-4. BCD Register Display
2-5. Octal Register Display
2-6. Octal Fraction Display
2-7. BCD Magnitude and Sign Display
2-8. Segment "a" Logic

3. KEYBOARD DATA INPUT

3-1. Keyboard BCD Coding
3-2. Keyboard to Display
3-3. Data Entry and Octal Display
3-4. BCD Display, Keyboard and Register

4. RANDOM ACCESS MEMORY

4-1. Random Access: Write-Read
4-2. Memory, Register and Display
4-3. Automatic Address Memory

5. REGISTER PRESETS AND DECODING

5-1. Register and Decoder

6. MULTIPLEXING AND DATA TRANSFER

6-1. Preset Instruction Register
6-2. Counter Preset & Data Transfer
6-3. Multiplexer

7. ADDER AND OCTAL ARITHMETIC

7-1. Full Adder
7-2. Adder with Memory
7-3. Subtraction with Memory
Octal Arithmetic
7-4. Bit Delay Master Slave
7-5. Master - Slave Triggering
7-6. Serial Register and Adder
7-7. Instruction Arithmetic Unit
7-8. Programmed Memory and Arithmetic Unit
7-9. Line Frequency Counter

8. AUTOMATIC DIGITAL COMPUTER

Block Diagram
Operations
Program Instructions
8-1. Timing & Control
8-2. Arithmetic Unit
Logic Troubleshooting
8-3. Memory and Address Control
8-4. Automatic Digital Computer

9. ARITHMETIC AND MEMORY SYSTEM

9-1. Arithmetic Unit
Instruction Tables
9-2. Memory Unit
9-3. Test Maintenance Programs
Troubleshooting

10. PROGRAMMING AND FLOW CHARTS

Programs
Symbolic Program
10-1. Machine Language
10-2. Flow Charts

11. ITERATIVE PROGRAM LOOPS

11-1. Multiplication
11-2. Iterative Loops
11-3. Decision Loops

12. PROGRAMMING TECHNIQUES

12-1. Command Variations
12-2. Bit Selection
12-3. BCD to Binary Conversion
12-4. Binary to BCD Conversion
Decimal Computation
12-5. Overflow
Subroutines
Computer Languages

13. PRINTED CIRCUITS AND INTEGRATED CIRCUITS

13-1. Integrated Circuits
P.C. Card

14. DIGITAL OUTPUT REFRESHER

14-1. Data Refresher

15. DIGITAL COMMUNICATIONS

Pulse Code Modulation
15-1. Transmit Data Register
Pulse Amplitude Modulation
Frequency Shift Keyer
PDM Modulator
15-2. PDM Modulator - Transmitter
15-3. PDM Decoder
PDM Modem
15-4. Remote Display System

16. NUMERICAL CONTROL SYSTEM

16-1. Sequential Memory
16-2. Numerical Control Graphics Display

17. ELECTRONIC MUSIC

17-1. Tone and Octave Generation
17-2. Digital Sequence Control
17-3. Keying System
Rhythm Counter
17-4. Rhythm System
17-5. Base Rhythm
17-6. Tuning
17-7. Rhythm Controller

18. DIGITAL MODEM

Digital Pulse Width Modulation
Digital Phase Modulation
Synchronization
ASCII 8 Bit Code
ASCII Display Connection
18-1. Phase Modulation Transmitter
18-2. Phase Modulation Receiver
PCM Carrier Frequency Modulator
PCM Detector
Photocell Signal Conditioner

19. ADDITIONAL PROGRAMMING METHODS

Retrieval
Location
Search
Sort
19-1. Location Problem
19-2. Scaling
19-3. Double Precision
Floating Points
19-4. Real-Time Timing
19-5. Square - Root Algorithm
19-6. Problem Solving

20. INPUT-OUTPUT INTERFACE

Digital Voltmeter Display

20-1. 0.5 Volt Resolution
20-2. 0.25 Volt Resolution
20-3. Digital Voltmeter Instrument
Computer Peripheral Equipment
Remote Terminal
20-4. Remote Output Display Terminal
20-5. Remote Input - Output Terminal
Interactive Terminal
20-6. Calculator Program

21. ALPHANUMERIC DISPLAY SYSTEM

21-1. Alphanumeric CRT Display

22. GRAPHICS CRT DISPLAY

22-1. Graphic Format Display

23. MORSE CODE KEYER

23-1. Code Keyer Logic

24. AUTOMATIC KEYBOARD ENTRY

24-1. Automatic Keyboard Entry

25. DIGITAL FREQUENCY SYNTHESIZER

25-1. Digital Frequency Synthesizer
25-2. Phase Comparator
25-3. Frequency Comparator
25-4. VCO Oscillator
25-5. Phase-Locked-Loop/open loop
25-6. Phase-Locked-Loop/closed loop

COMPUTER CONFIGURATION EXPERIMENT:

TYPE:

Stored Program Single Address,
Serial Arithmetic, Random Access Memory
Automatic Instruction and Operate Cycles,
Continuous or Single Step Operation;
Manual, High, and Low Speed Clock.

FORMAT:

8 Bit Words,
Data: 8 bits including sign
Instruction: 5 bits address,
3 bits operation
32 Memory Words.

INSTRUCTIONS:

Load (and Input),
Add, Subtract,
Shift Right, Jump,
Store (and Output),
Branch on Negative, Halt.

INPUT - OUTPUT:

12 Input Keyboard,
3 Digital Octal Display,
Digital Switches & Lamps
for Control and Address
and Mode Display.

Ed-Lab

EXPERIMENT MANUAL

ADVANCED DIGITAL SYSTEMS

INTRODUCTION AND COUNTERS EXPERIMENT NO. 1

OBJECTIVE:

This introduction summarizes the components on the Computer Memory and Display Lab (Ed - Lab 900) and describes their operation. Experiments are performed with the Counters to demonstrate the trainer operation.

GENERAL:

The Ed - Lab 900 contains a wide variety of Computer Input, Ouput and Logic elements to provide a comprehensive training capability. A number of experiments can be performed using the Ed - Lab 900 alone, however, when Ed - Lab 900 is used together with an Ed - Lab 700 Computer Systems Lab, a wide variety of experiments can be performed in computer technology, programming, circuits, digital communications, skill projects, and logic troubleshooting.

The computer elements on the Ed - Lab 900 are:

1. Hexadecimal Display Output* - 3 digits plus decimal points - (LEDs)
2. Keyboard Input - 16 keys* - (Elastomeric Contacts)
3. Memory - 256 bits Random Access Memory - (MOS)
4. Arithmetic Shift Register - 8 bits
5. Instruction Shift Register - 8 bits
6. Program Counter - 8 bits
7. Digital Multiplexer - 8 bits
8. Digital Decoder - 3 to 8 bits
9. Full Adder - Sum and Carry
10. Bit Delay - Flip Flop Carry Memory
11. Complement - 4 bits (4 NOT Elements)
12. Display Lamps - 5 bits - (LEDs)
13. Input Switches - 5 binary
14. Pulse Switch - 1 Momentary Pulse
15. Dual In - Line Integrated Circuit Socket - 14 and 16 pins type
16. Printed Circuit Socket - 22 pins standard
17. Tie - Points for Interconnections - 3
18. Ground Bus Line
19. Line-Frequency Clock

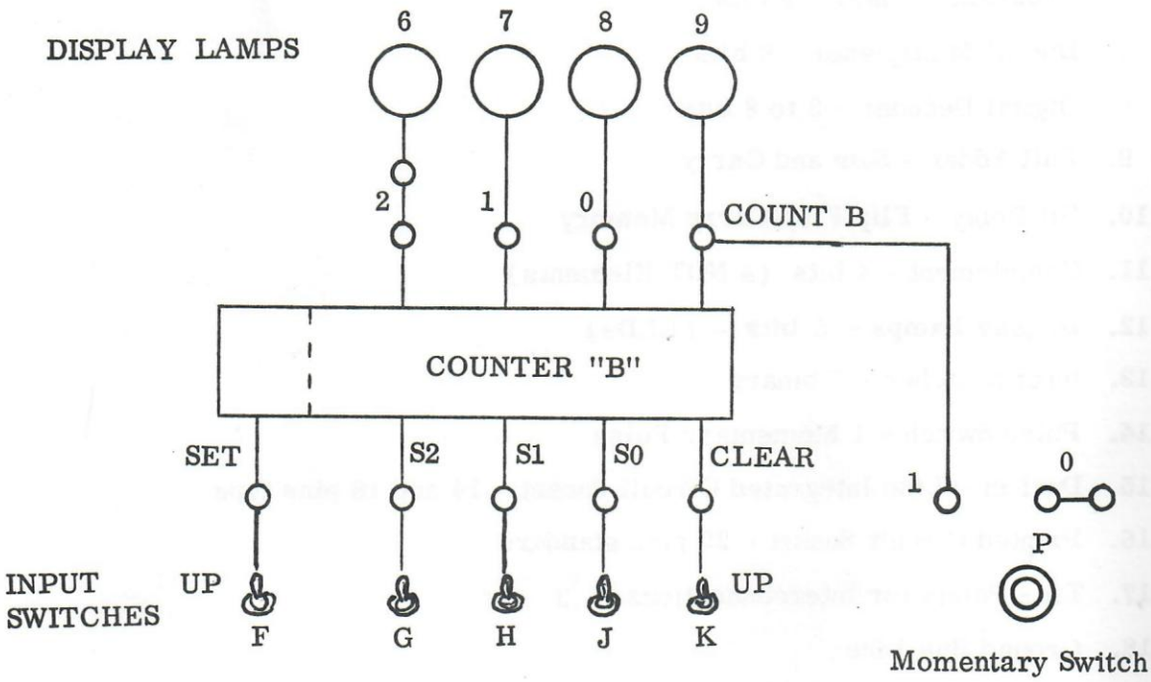
* ON EARLIER UNITS WITH 12 KEY KEYBOARD, THE DISPLAY IS OCTAL & DECIMAL ONLY.

At the end of this first experiment, a detailed operating description is given for each element, for convenience. It is not, however, necessary to review these elements and their operations at this time. Each element is introduced in a subsequent experiment together with test demonstrations and applications.

This first experiment demonstrates the COUNTER operation (PROGRAM COUNTER "P" AND COUNTER "B"), the Binary Input Switches (F, G, H, J, K), the Momentary Switch (P) and the Binary Display Lamps (6, 7, 8, 9, 10). The letters for the switches follow those letters on the Ed - Lab 700 (A, B, C, D, E) and the Binary Display Lamp numbers also follow those on the Ed - Lab 700.

DESCRIPTION (COUNTER B):

The counters P and B have a total of 8 bits; with 2 separate count inputs; all 8 outputs; 8 presets; and one clear and one set input. The B counter is a 3-bit counter with the "COUNT B" input and outputs marked 0, 1, 2. A "0" or ground on CLEAR sets the bits to "0". A "0" on SET sets the outputs as determined by the inputs on S0, S1, and S2 (S = SET). The input on S0, S1, and S2 is "0" when grounded and "1" when "open" or not connected or when connected to an element with a binary "1" output. The count changes when "COUNT B" goes from "1" to "0".



Experiment 1-1
BIT COUNTER "B"

PROCEDURE:

1. Wire the logic. With switches F and K UP, operate switch P and observe the count on Display Lamps.

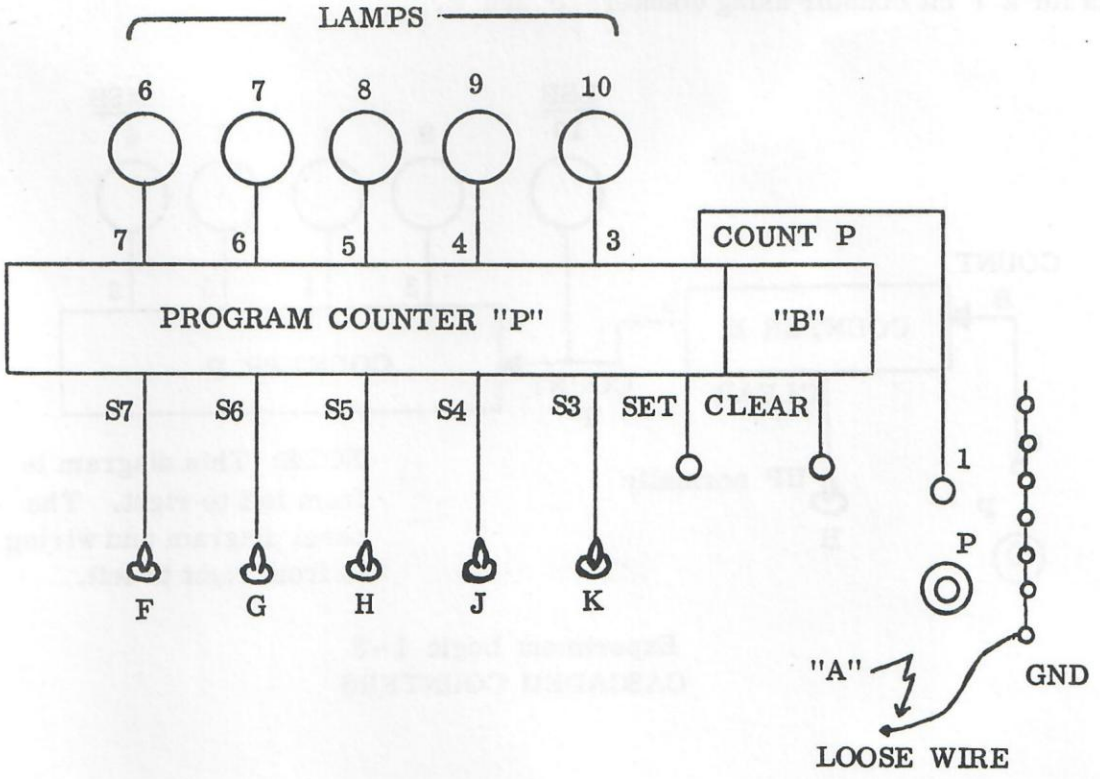
2. Set switch K down - then up - to CLEAR.

3. Set switches G, H, J for a binary number.

4. Set switch F down - then up - to set in number G, H, J.

DESCRIPTION (COUNTER P):

Counter P has 5 bits denoted as bits 3, 4, 5, 6, 7. Set up this counter logic.



Experiment 1-2
PRESET COUNTER "P"

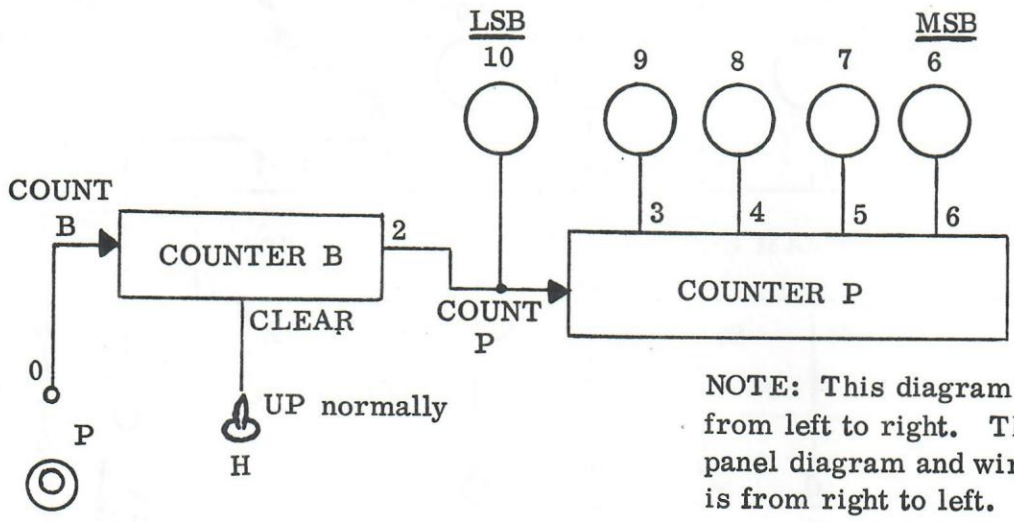
PROCEDURE:

Wire the Logic. Count using switch P. Use the Loose Lead "A" from GND to CLEAR or SET the counter. Set in various numbers. Enter the data below:

S7	S6	S5	S4	S3	CHECK

DESCRIPTION (Cascaded Counter):

The LOGIC DIAGRAM layout does not have to correspond to the panel layout. The important consideration is that the "markings" on the diagram correspond to the proper wired connections. The logic diagram below is for a 7 bit counter using counters B and P.



NOTE: This diagram is from left to right. The panel diagram and wiring is from right to left.

Experiment Logic 1-3
CASCADED COUNTERS

Wire the logic. Count via pulse P. Observe that the count changes once for every "four" pulses. Switch H down - then up - to clear the counter.

QUESTIONS:

1. What is the maximum number of different counts for counter P?

2. What is the maximum number of different counts for counter B?

3. What is the maximum number of different counts for a counter consisting of counter B followed by counter P?

4. What is the highest count for counter P?

SUMMARY OF ELEMENTS: (The student may skip this summary and continue to Experiment No. 2.)

1. HEXADECIMAL DISPLAY:* The display consists of three seven - segment characters plus decimal points. Binary inputs 1, 2, 4, 8 are decoded to display the number. An unconnected lead on a jack is a "1". The Binary codes for 0 thru 9 are properly displayed. The Binary codes for 10 thru 15 display the letters A through F. A "0" at the "P" jack results in a decimal point being displayed. A "0" at BL turns ON the display. A "0" at "-" Lights the center segment of the left digit for a minus sign. A "0" at any of the segments marked ⊟ results in that segment being illuminated, making any number and numerous letters such as A C E F H J L P available on the right digit.

2. KEYBOARD: The 16 Key Keyboard is coded to generate Hexadecimal outputs 8, 4, 2, 1, switched to COM (common). The common is normally connected to "0" (ground) so that when a key is depressed, the code is a "0" at the coded locations. Thus, number six is 0110. The "S" (strobe) signal is the AND of these signals and thus becomes "0" when any key is depressed (with contact bounce). The "DS" signal is a "delayed" strobe pulse which becomes "0" for SET purposes without "contact bounce". Because the binary coded "one" is a "ground" signal, an Inverter (NOT/COMPLEMENT) is needed on each line, when using the keyboard to preset a counter or a shift register. The delayed strobe (DS) is used on the register SET line.

3. MEMORY: The memory is an MOS solid - state Random Access Memory (RAM) with 256 bits. The 8 address lines are denoted as 0, 1, 2, 3, 4, 5, 6, 7, on the X Decode and Y Decode inputs. DATA INPUT is a Binary "1" or "0". The Memory matrix will only operate when the SELECT line is "0" or ground. (When many matricies are used, the address is connected to all units and the selected unit is connected with a select = 0.) The DATA OUTPUT, A, presents the memory data. To read, a "0" or ground is connected to READ = 0; to write a "1" or "open" must be connected to WRITE = 1. The SELECT, ADDRESS, and DATA - IN must all be set before the WRITE = 1 is applied

4. ARITHMETIC REGISTER "A": This unit is an 8 bit shift register with input DATA - IN at bit 7 and shifted right to bit 0. Bits marked 0 to 7 are the outputs, and N7 is the complement of the most significant bit (MSB) (7). This is used when the MSB represents a SIGN because a "ground" is required to activate the "minus" (-) display input. Inputs S0 thru S7 are for presets. The CLEAR is operated by a "0" and SET 0 - 2 ("0") sets bits 0, 1, 2, while SET 3 - 7 sets bits 3, 4, 5, 6, 7. When entering new data, a CLEAR is first applied and is then followed by a SET. The SET will only set an output "1". The shift is activated by an input change from "1" to "0".

* on earlier units the display is decimal and the BL terminal connected to Ground BLANKS the display. The LT terminal turns ON all segments of the decimal display.

5. INSTRUCTION REGISTER "I": This 8 bit shift register is the same as register "A" with a single SET input for all 8 bits.

6. PROGRAM COUNTERS "P" AND "B": Counter B is a 3 bit counter with a "COUNT B" input (counts on "1" to "0"). Outputs are 0, 1, 2. Set inputs are S0, S1, S2. The CLEAR is activated by "0". The SET is activated by "0".

Counter P has a "COUNT P" input for the 5 bit counter (Outputs 3, 4, 5, 6, 7). The preset inputs (S3 thru S7) are activated via the same CLEAR and SET inputs used with counter B.

7. DIGITAL MULTIPLEXER (M): The 8 bit digital multiplexer is internally connected to the COUNTERS (P and B) and the INSTRUCTION REGISTER "I". When "SELECT P = 0" is connected to "0", the "MUX" (M) output corresponds to the COUNTER outputs. When "SELECT P = 0" is open or at a "1", the MUX output corresponds to the REGISTER "I" output. The MUX is generally used when addressing the memory and alternating between the COUNTER address and the INSTRUCTION address.

8. DIGITAL DECODER: The decoder accepts the 3 binary bits (A, B, C) and generates the 8 decoded outputs (N0 to N7). "N" refers to NOT since all outputs are normally "1" and only the one selected output is a "0". The EN is an enable line which must be a "0" to enable any of the outputs to be decoded to "0".

9. FULL ADDER: The Full Adder has standard A, B, C inputs with Sum (S) and Carry (K) outputs. Signal (NS) is the complement of S. Enable signals EA and EB are AND gate inputs to control inputs A and B. When EA is "0", then input A is disabled or taken as "0". EB disables (AND gate) input B.

10. BIT DELAY: The Bit Delay is a type D Flip - Flop with Outputs Q and NQ; Presets NR (reset = 0) and NS (set Q = 1 when NS = 0); the Input Steering D input is transferred to Q when a "1" to "0" step is applied to the trigger T.

11. COMPLEMENT: The four bit complement unit consists of four NOT elements with input A, output NA, etc.

12. DISPLAY LAMPS: The 5 display lamps are ON with a "1" or no input. A "0" input results in an OFF lamp.

13. INPUT SWITCHES: The 5 input switches present a "1" output when UP, and a "0" (ground) output when DOWN.

14. PULSE SWITCH: The Pulse Switch P is a momentary switch input which sets an internal flip - flop generating one sharp pulse without bounce. The "1" terminal is normally a "1" and the "0" is normally a "0".

15. DUAL IN - LINE SOCKET: A 16 pin dual - in line integrated circuit socket will provide capability for a 14 or 16 pin I. C. Pin 1 is on the lower left hand side as marked. The 16 pins are connected to pins 1 - 16. When a 14 pin I. C. is used pin 8 is on the 10/8 jack, etc. The +P jack provides +5 volts via a series 33 OHM RESISTOR. The -N jack provides -9 VOLTS via a series 100 OHM RESISTOR. The terminal marked GND furnishes "ground" which is the same signal provided on the GND bus.

16. PRINTED CIRCUIT SOCKET: The printed circuit socket has contacts on 2 sides with 22 contacts on each side. The contacts on the left side (1 on top) are connected to the 22 marked panel jacks.

17. TIE - POINTS: The tie - points T11, T12, and T13 are used to provide multiple connections.

18. GROUND BUS LINE: The tie - point T14 provides a grounding (binary "0") tie point.

OBJECTIVE:

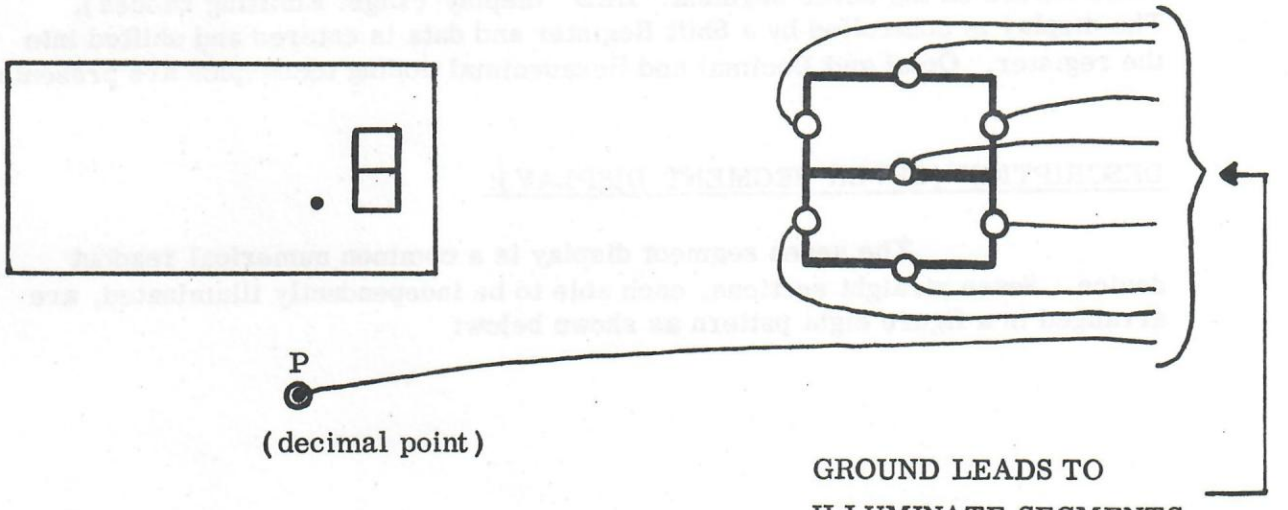
This experiment describes the display of all the numbers and some letters on the seven segment "LED" display (Light Emitting Diodes). The display is controlled by a Shift Register and data is entered and shifted into the register. Octal and Decimal and Hexadecimal coding techniques are presented.

DESCRIPTION (SEVEN SEGMENT DISPLAY):

The seven segment display is a common numerical readout device. Seven straight sections, each able to be independently illuminated, are arranged in a figure eight pattern as shown below:

SEVEN SEGMENT DISPLAY

The Ed - Lab 900 has three display characters. Each segment of the right hand display digit is controlled by grounding and ungrounding the points on the schematic of the seven segment display. This is illustrated on the following page.



Experiment 2 - 1

SEVEN SEGMENT DISPLAY

PROCEDURE (SEVEN SEGMENTS):

1. Ground the seven wires to the segments of the display and one wire to the decimal point as shown in Figure 2 - 1.

2. Light the segments selected to display the numerals 0 thru 9 and as many English Letters as you can. List all the letters that can be formed using the 7 segment display. Letters which are the same as numerals are not valid (i.e. 8 and B look alike in 7 segment). In using the 7 segment display the number 6 is 6 and the letter b is b.

3. Produce the minus sign and the equal sign.

4. Draw the numbers and letters as they appear on the 7 segment display.

NUMBERS	Display	LETTERS	Display
0		A	
1		b*	
2		C	
3		d	
4		E	
5		F	
6*		H	
7		J	
8		L	
9		P	
		U	
		=	
		-	

* Remember 6 = 6 and b = b .

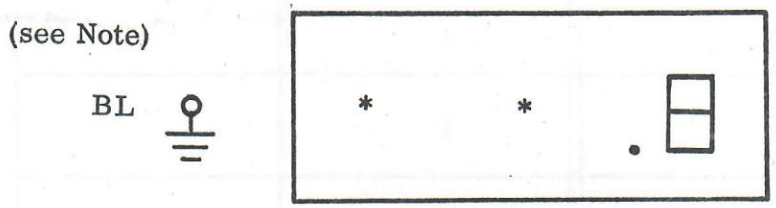
QUESTIONS:

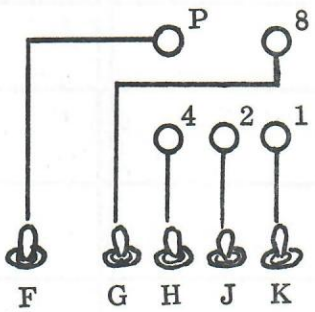
1. How many segments are needed to produce all the numbers and letters of the alphabet?

2. How would these segments be arranged?

DESCRIPTION (BINARY CODED DECIMAL & HEXADECIMAL)

The switches K, J, H, and G are connected to the 1, 2, 4, and 8 inputs to a display control logic section. Switch F is connected to the decimal point (P) of the same segment. This is shown in Figure 2 - 2 below.





NOTE: FOR UNITS WITH 12 KEY KEYBOARD, The BL pin must be removed from GROUND to turn the DISPLAY ON.

* For units with 16 key keyboard the two left digits will display FF when inputs are not connected.

Experiment 2 - 2
BINARY - CODED - DECIMAL DISPLAY

The four switches G, H, J, and K control the binary bits for a 4 bit binary number. The display logic decodes this binary number into a decimal digit from 0 to 9. The corresponding characters above 9 are represented by the letters A b C d E F for 10, 11, 12, 13, 14, 15 respectively.

If we use only the binary input codes 0-9 then the system is called Binary Coded Decimal.

When we permit the use of all binary codes 0 through 15 the system is Hexadecimal and the output display will present the letters A through F for the numbers 10 through 15.

PROCEDURE (BCD DISPLAY CONTROL):

Connect the switches to the display inputs as shown in Figure 2-2. Manipulate the switches through the binary sequence from 0 to 15 with and without decimal point. Enter the data in Table A. Fill - in all combinations of inputs.

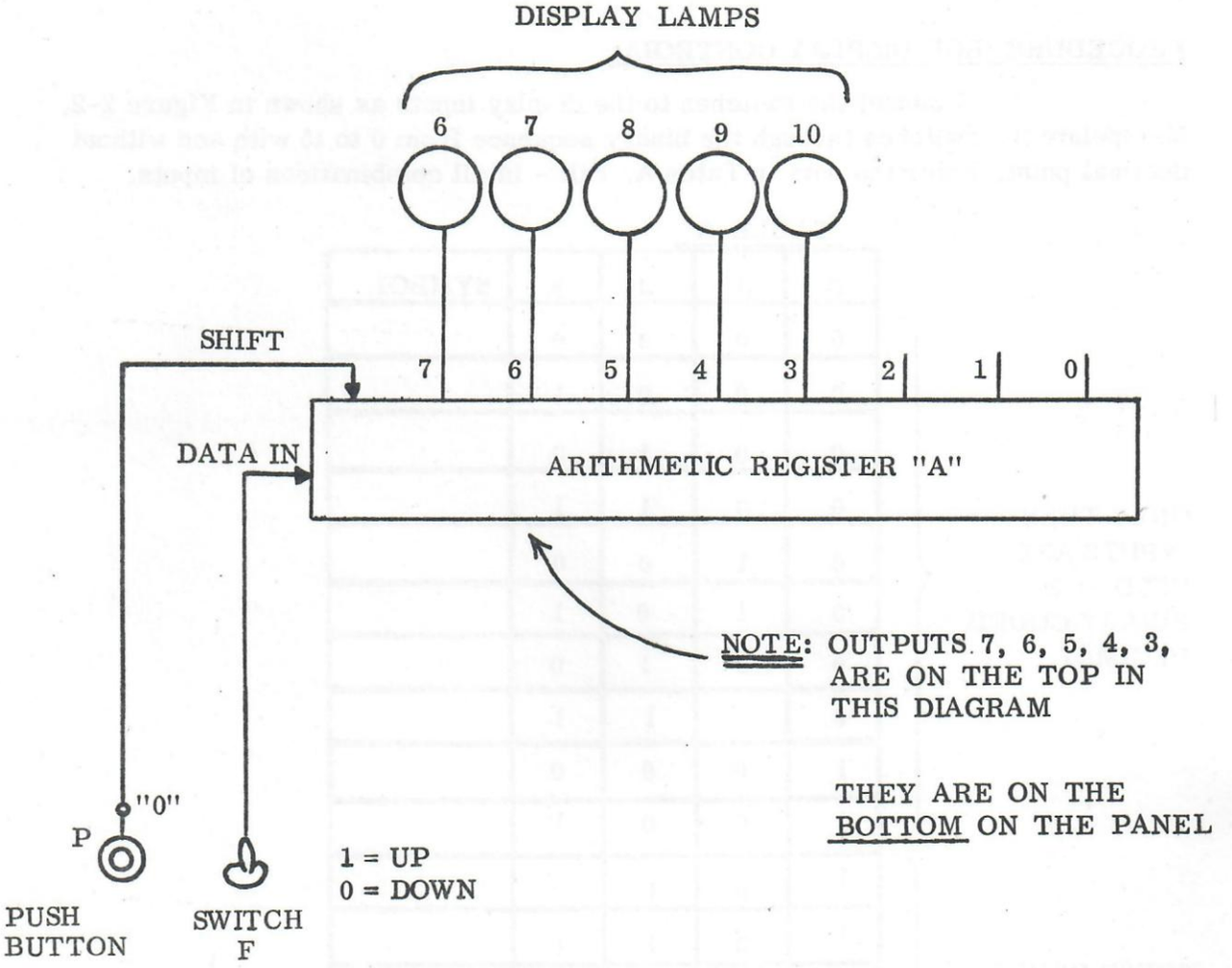
TABLE A

	G	H	J	K	SYMBOL
ONLY THESE INPUTS ARE USED FOR BINARY CODED DECIMAL	0	0	0	0	
	0	0	0	1	
	0	0	1	0	
	0	0	1	1	
	0	1	0	0	
	0	1	0	1	
	0	1	1	0	
	0	1	1	1	
	1	0	0	0	
	1	0	0	1	
THESE CODES ARE USED IN HEXADECIMAL*	1	0	1	0	
	1	0	1	1	
	1	1	0	0	
	1	1	0	1	
	1	1	1	0	
	1	1	1	1	

* NOTE: Units with 12 Key Keyboard will not generate the correct Hexadecimal output display for the input codes 10 through 15.

DESCRIPTION (SHIFT REGISTERS):

The Arithmetic Register "A" is a parallel output / serial or parallel input shift register. This experiment demonstrates the operation of the serial input and parallel output features. The DATA INPUT to the register is set by "Switch F" and the register is SHIFTED by the push button output. This system is shown below.



Experiment 2 - 3
SHIFT REGISTER

The shift register is connected to the display using 5 different coding methods.

(1) The first five bits are connected in binary (Experiment 2-3, above).

(2) Each set of 4 bits is used to display a BCD (binary coded decimal) decimal number, 0 through 9 and a Hexadecimal Number 0-F.* (Experiment 2-4.)

(3) A straight OCTAL (base 8) connection is used with the incomplete octal digit (2 bits) on the left; this is generally used when the decimal point is to the right. (Experiment 2-5.)

(4) OCTAL coding when the number is represented as a fraction (less than 1). (Experiment 2-6.)

(5) The BCD number is given as a magnitude plus sign (sign is represented by the first-left-bit). (Experiment 2-7.)

PROCEDURE:

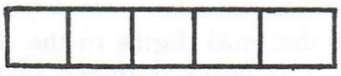
Wire the logic shown in Figure 2 - 3. Switch F is used to specify the input data bits and the push-button is used to shift the register. Enter all zeros into the shift register by placing switch F down and clocking the register using Switch P at least 7 times. Place Switch F UP and shift the register once. Place Switch F down and pulse (P) the single 1 through the register. Lamp 6 is the most (5th) significant bit and lamp 10 is the least significant bit.

Shift a single "1" bit down the line as described above.

By varying the input Switch F, shift-in the binary number for 26 (11010).

Enter the number, in the boxes below, as read-out from the display:

Display Lamps 6 7 8 9 10



QUESTIONS:

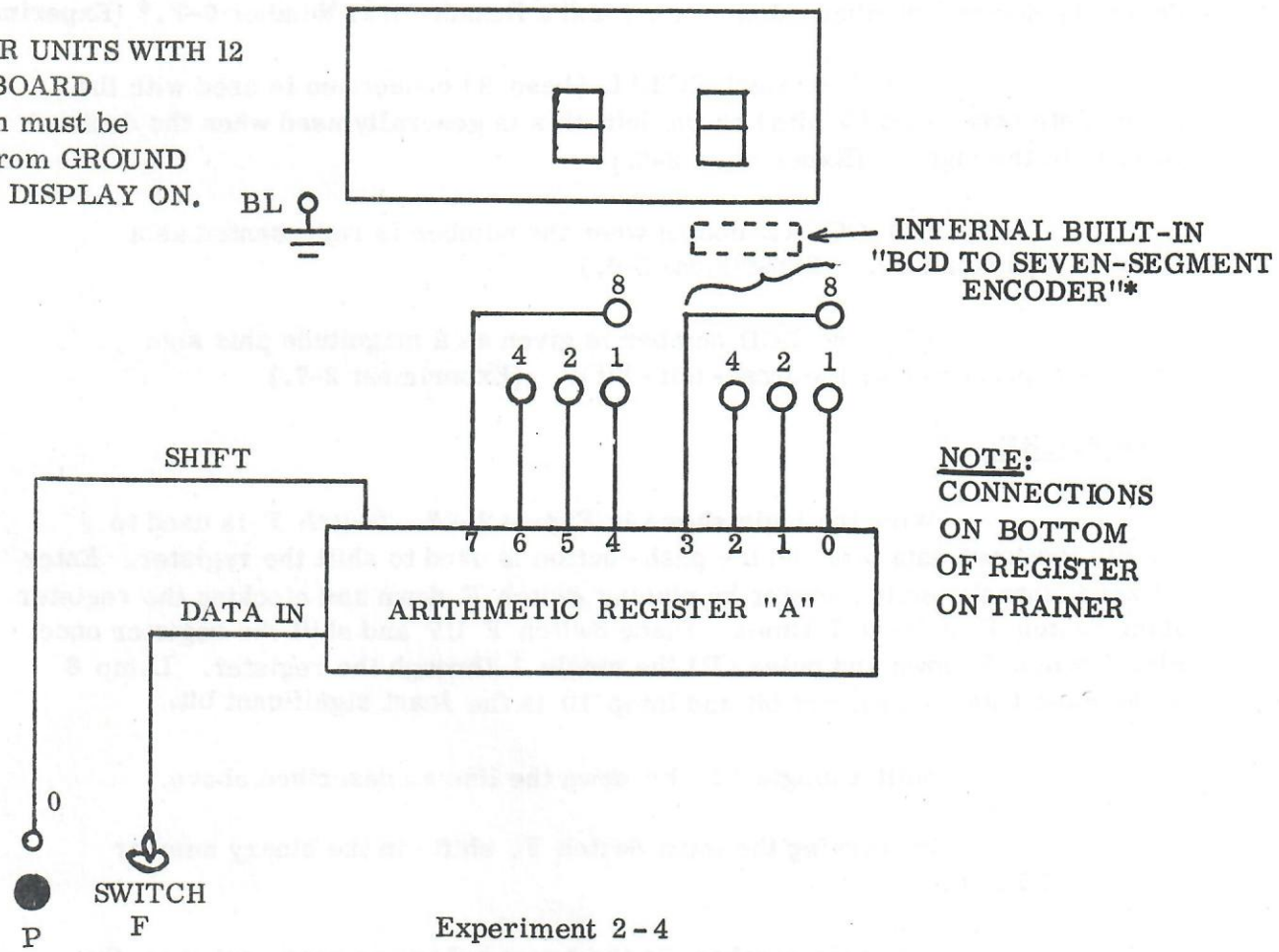
3. The shift register appears to be shifting to the right. Draw the schematic for the interconnections to make it appear to shift to the left.

* Hexadecimal numbers cannot be represented on earlier units which have a 12 key keyboard.

PROCEDURE (BINARY CODED DECIMAL):

Remove the 5 wires connected to the display lamps and wire the shift register outputs to the display as shown below:

NOTE: FOR UNITS WITH 12 KEY KEYBOARD
The BL pin must be removed from GROUND to turn the DISPLAY ON.



Experiment 2-4
BCD REGISTER DISPLAY

The two right hand decimal digits in the display are used as the readout from the shift register. The number in the shift register is in binary-coded-decimal form. Register outputs 0-3 drive the units digit; 4 - 7 are used to drive the tens digit. Shift the indicated decimal numbers into the shift register in binary coded decimal and check the decimal readout. Enter the data in Table B.

Using the same circuit shift in the Hexadecimal numbers indicated in Table B1 and record the Output Display.

* On page 16 of this Chapter is Experiment 2-8 which demonstrates how a segment line is decoded from a BCD number.

TABLE B (BINARY CODED DECIMAL)

TEST	DATA IN BINARY	Display
1	FIRST ↓ 00110001	31
2	01100111	
3	10010101	
4	11000101	
5	10001010	
6	01100010	
7	01101111	

TABLE B1 (HEXADECIMAL)

TEST	DATA IN BINARY	Display
1	10100110	
2	01111111	
3	10101011	
4	11001001	
5	00111000	
6	01001100	
7	11110000	

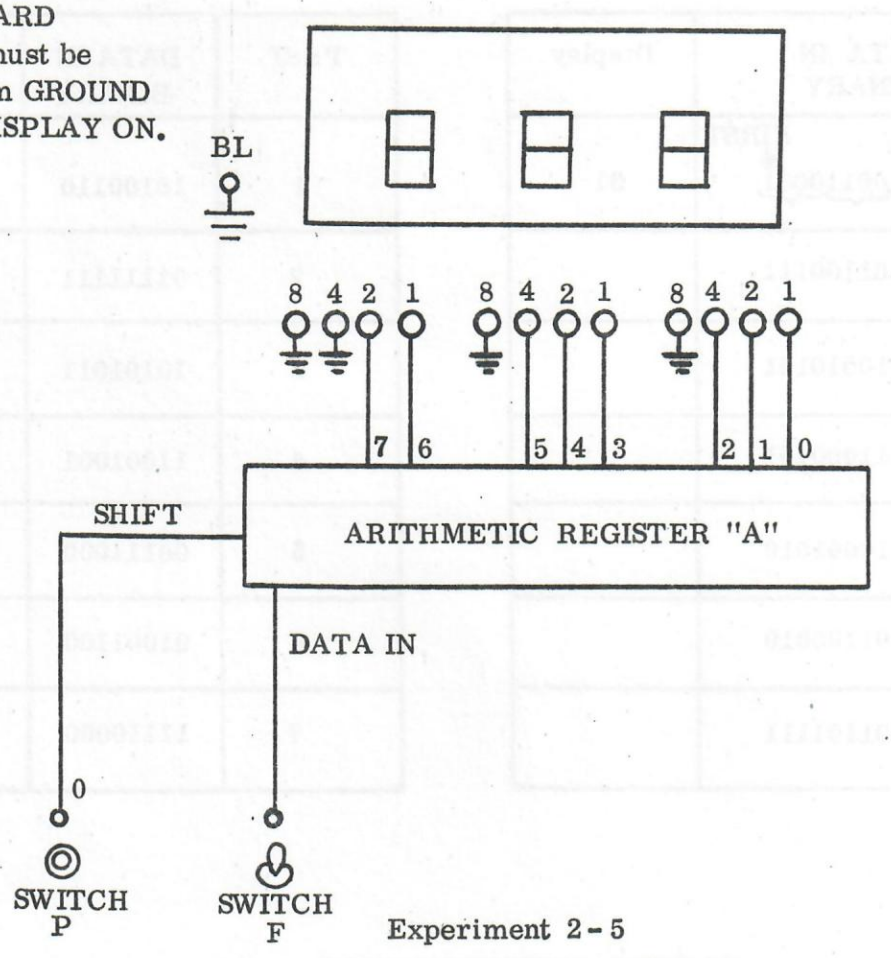
QUESTIONS:

4. If the readout is 62 and a "0" enters from the left at the next shift, what number will appear on the readout after the shift?

PROCEDURE (OCTAL CODE):

Change the display wiring as shown below:

NOTE: FOR UNITS WITH 12 KEY KEYBOARD
The BL pin must be removed from GROUND to turn the DISPLAY ON.



Experiment 2-5
OCTAL REGISTER DISPLAY

This corresponds to 8 bit octal coding. The most significant octal digit (bits 6 and 7) varies from 0 to 3 because it uses only 2 binary bits. The other two digits may vary from 0 to 7. Thus the right hand "octal digit" is 1's digit; the next one (middle) is 8's digit; and the left hand digit is the 64's digit. This is then octal coding and counting in the base 8. Shift-in the binary bits (least significant bit first)

to produce the required octal number, then calculate the decimal equivalent. Enter all data in Table C below:

TABLE C

TEST	OCTAL NO.	BINARY (fill in)	DISPLAY	DECIMAL EQUIV.
1	013	00 001 011 (FIRST)		
2	206			
3	127			
4	035			
5	062			
6	123			
7	321			
10	149			
11	222			
12	325			

Note (TEST 1):

DECIMAL EQUIV. = (64 x 0) + (8 x 1) + (1 x 3)

QUESTIONS:

5. What is the highest number that can be displayed?

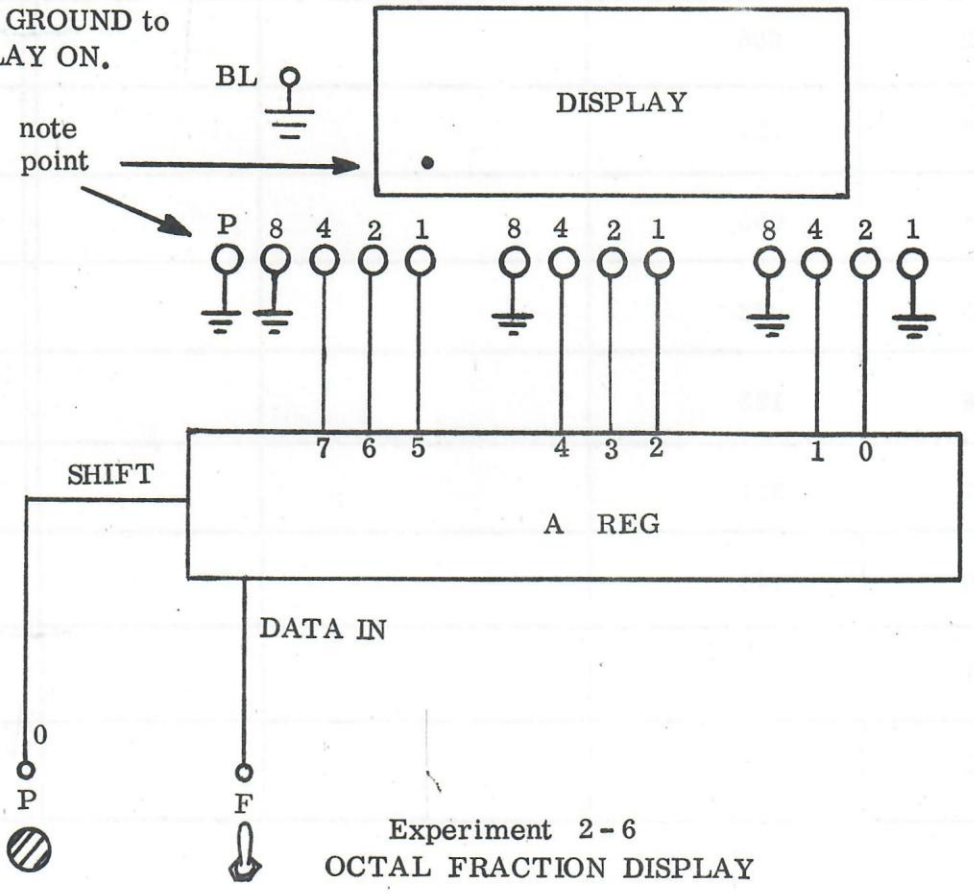
6. What is the advantage of an octal coded system over a binary coded decimal system when using binary logic?

PROCEDURE (OCTAL FRACTION):

With the decimal point on the left of the number, the lefthand digit must be a full octal number (3 bits) and the right hand digit remains with only two bits.

Change the wiring to the display as shown below:

NOTE: FOR UNITS WITH 12 KEY KEYBOARD
The BL pin must be removed from GROUND to turn the DISPLAY ON.



Experiment 2-6
OCTAL FRACTION DISPLAY

Use Table D and calculate the decimal equivalents (left digit 1/8's, next 1/64's, next 1/512's).

TABLE D

TEST	OCTAL NO.	BINARY (fill in)	DISPLAY	DECIMAL EQUIV.
1	.232	2 3 2 010 011 010 ← missing		
2	.726			
3	.776			
4	.621*			
5	.002			

*Note: This must be rounded off to the nearest number possible.

QUESTIONS:

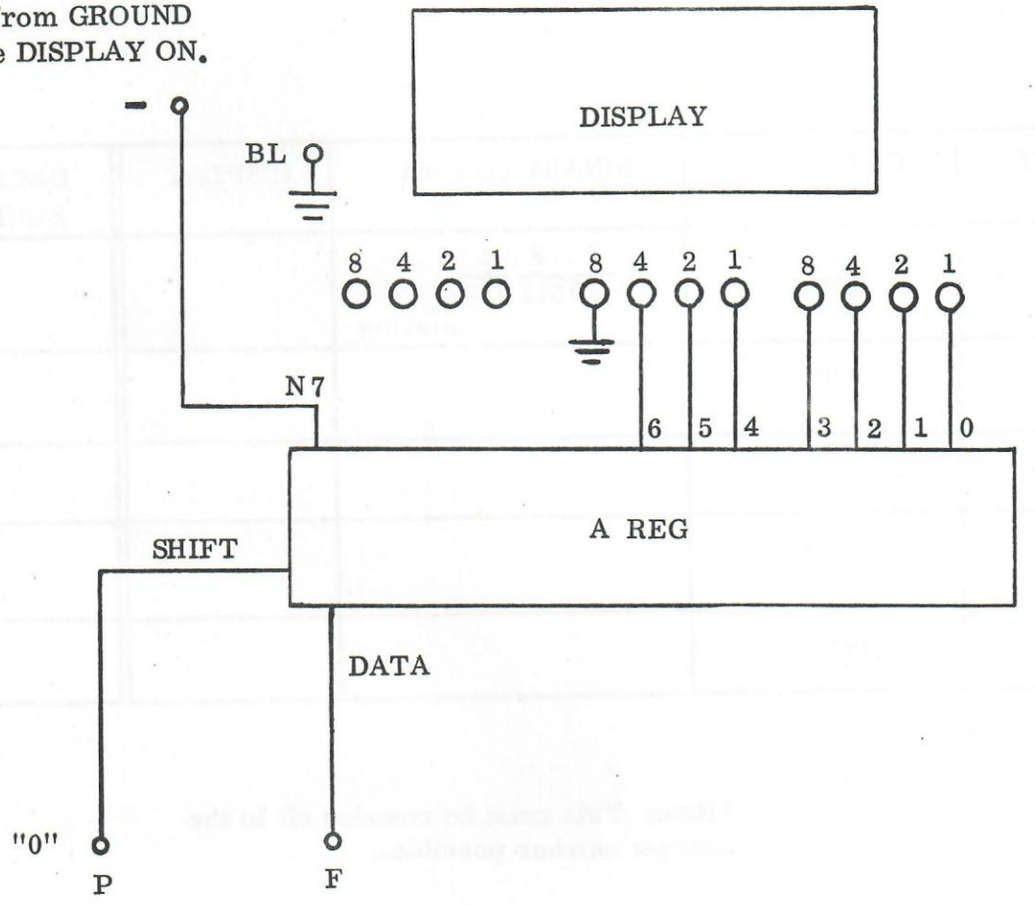
7. What is the maximum number that can be displayed on the octal fractional readout?

8. What is the resolution of the display (smallest increment)? What is it in octal and in decimal?

PROCEDURE (BCD WITH SIGN):

Change the display wiring as shown in the circuit below.

NOTE: FOR UNITS WITH 12 KEY KEYBOARD
The BL pin must be removed from GROUND to turn the DISPLAY ON.



Experiment 2-7
BCD MAGNITUDE/SIGN DISPLAY

The N7 bit is used to drive the minus sign (when bit 7 is a "1", N7 is "0"). The coding will use a sign plus a BCD number. Complete Table E, by shifting in the required binary number and recording the results.

TABLE E

TEST	BCD NO.	BINARY (fill in)	DISPLAY
1	- 14		
2	18		
3	39		
4	- 79		
5	+ 05		

QUESTIONS:

9. What is the maximum number that can be displayed with the BCD/SIGN format?

10. What is the smallest resolution?

11. What is the advantage of the BCD WITH SIGN format?

12. What is the best display and register format?

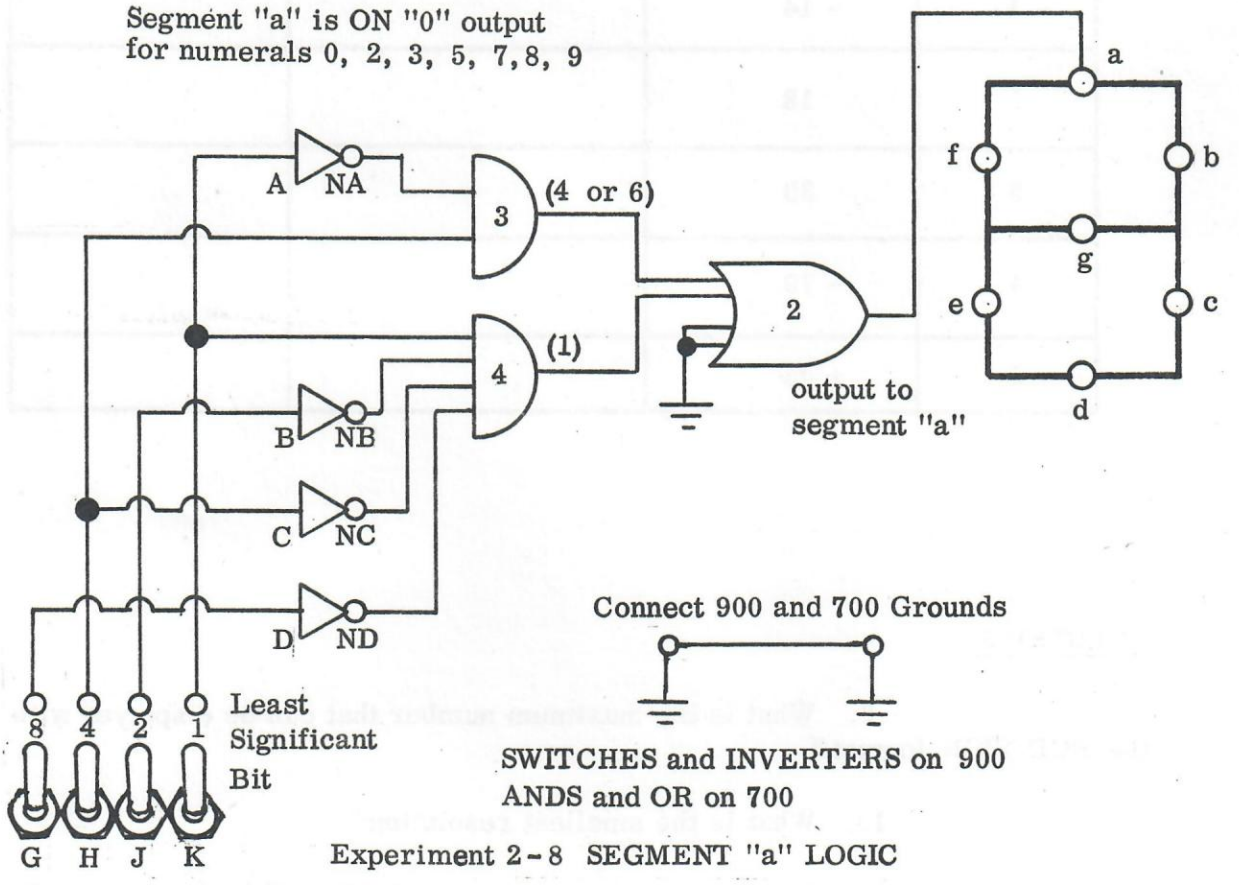
PROCEDURE (SEGMENT "A" LOGIC):

The Ed-Lab 900 has 3 built-in BCD to Seven Segment Encoders" for each of the display elements. Each Encoder is on a single integrated circuit which contains all of the necessary logic gating elements. A binary "0" or ground is required to illuminate a segment. The "a" segment (on top) is ON (binary 0) for the numerals:

(a = 0) for "2 or 3 or 5 or 7 or 8 or 9 or 0" = LIGHT ON

(a = 1) for 1 or 4 or 6

Segment "a" is ON "0" output for numerals 0, 2, 3, 5, 7, 8, 9



Experiment 2-8 SEGMENT "a" LOGIC

The logic shown above in Experiment 2-8 performs this function for segment "a" and uses both the Ed-Lab 900 and Ed-Lab 700.

1. Wire the Logic in Experiment 2-8
2. Operate the Switches G, H, J, K for all decimal binary numbers (zero to ten). Is the "a" segment ON at the required BCD number settings?_____
3. Draw the logic to illuminate segment "b". Test the logic. Is it correct?_____

KEYBOARD DATA INPUT EXPERIMENT NO. 3

OBJECTIVE :

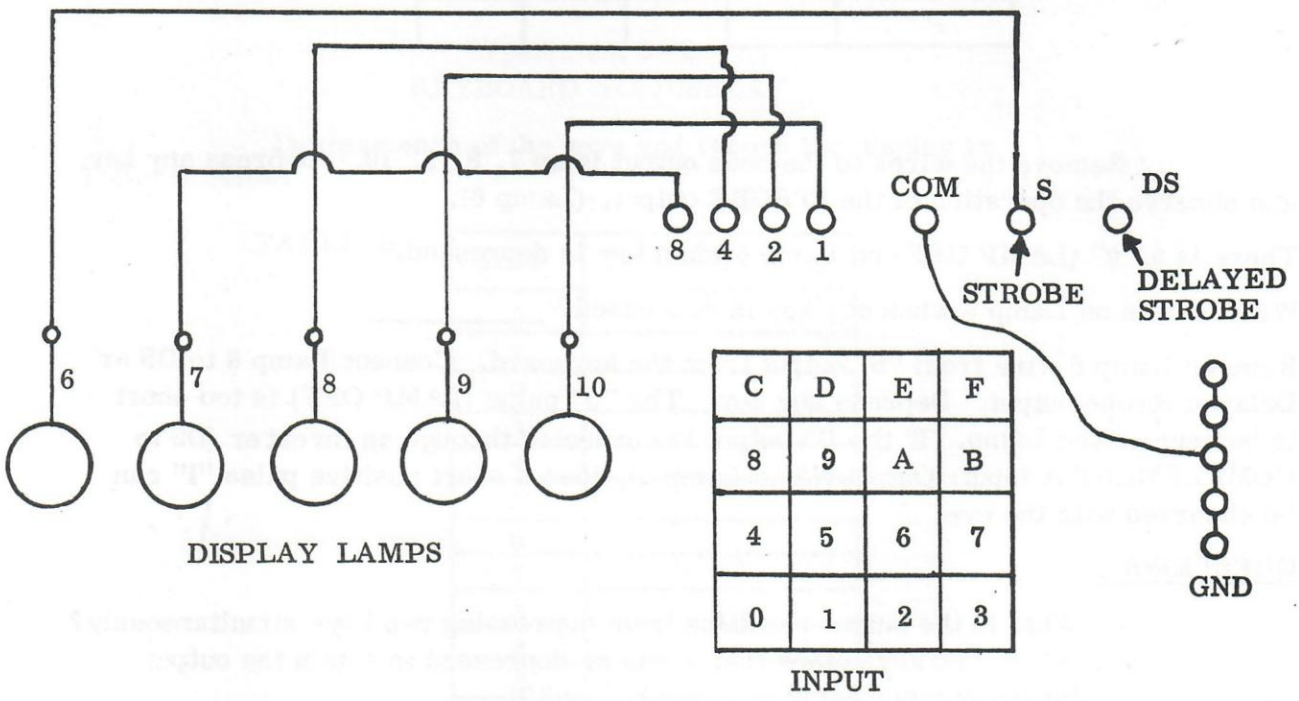
This experiment demonstrates the use of the KEYBOARD as an INPUT DEVICE, and the DISPLAY as an OUTPUT DEVICE. The Keyboard will then be used to load the ARITHMETIC REGISTER to drive the DISPLAY. This is done in BINARY and OCTAL coding, in BINARY CODED DECIMAL (BCD) and in HEXADECIMAL.

DESCRIPTION (KEYBOARD):

The Keyboard is activated by pressing the number keys on its face. These keys are coded to apply a binary number output to the output pins (8, 4, 2, 1,) of the keyboard. The signal connected to the common line COM is the signal switched to the coded binary outputs. If the common line is grounded and the #2 key is pressed, the ground will appear on the "2" output. All other outputs will be floating (or binary "1"). The STROBE is an AND of all outputs and is thus a "0" when any key is depressed. The DELAYED STROBE is a short pulse which occurs each time a key is depressed. This is used to SET data into the shift registers.

PROCEDURE :

Wire the circuit shown below.



Experiment 3 - 1
KEYBOARD CODING

NOTE: EARLIER TRAINERS HAVE 12 KEY KEYBOARD.

Depress each of the keys and record the output results in the table below:

KEY	8	4	2	1	← BIT CODE
	7	8	9	10	← LAMP #
0					
1					
2					
3					
4					
5					
6					
7					
8					
9					
A					
B					
C					
D					
E					
F					

Remove the wires to the code output lamp 7, 8, 9, 10. Depress any key and observe the operation of the STROBE output, (Lamp 6).

There is a "0" (LAMP OFF) on Lamp 6 when key is depressed.

What occurs on Lamp 6 when any key is depressed? _____

Remove Lamp 6 wire from "S" output from the keyboard. Connect Lamp 6 to DS or Delayed Strobe output. Depress any key. The "0" pulse (LAMP OFF) is too short to be seen on the Lamp. If the DS output is connected through an inverter (DS to COMPLEMENT A input: Output NA to Lamp 6), then a short positive pulse "1" can be observed with the eye.

QUESTIONS :

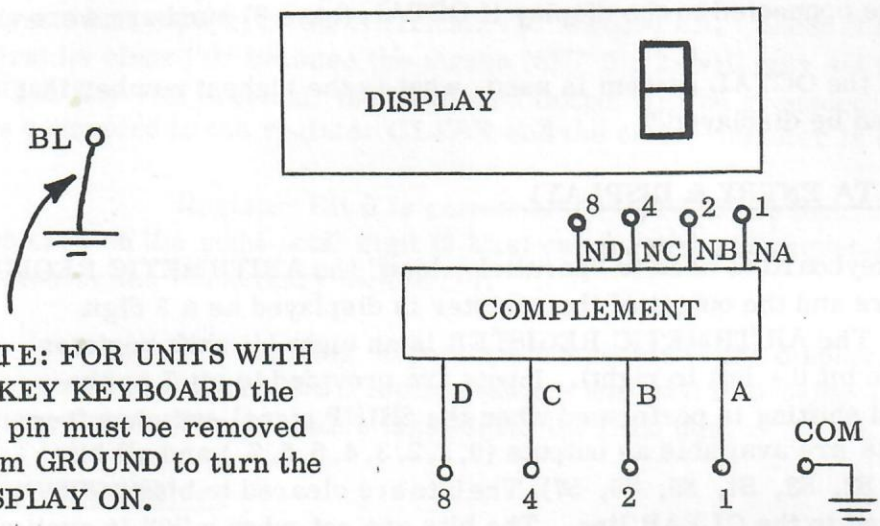
1. What is the output resulting from depressing two keys simultaneously? Which two keys other than 3 can be depressed to obtain the output binary number corresponding to key 3?

DESCRIPTION (KEYBOARD TO DISPLAY) :

The Keyboard output is connected to one digit of the display. Because the keyboard output is the binary complement of the number pressed, the display is driven through the COMPLEMENT (inverter) element.

PROCEDURE :

Wire the circuit shown below:



NOTE: FOR UNITS WITH 12 KEY KEYBOARD the BL pin must be removed from GROUND to turn the DISPLAY ON.

C	D	E	F
8	9	A	B
4	5	6	7
0	1	2	3

Experiment 3 - 2
KEYBOARD TO DISPLAY

Depress each of the keys and record the display in Table B below:

TABLE B

KEY	DISPLAY
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
A	
B	
C	
D	
E	
F	

QUESTIONS :

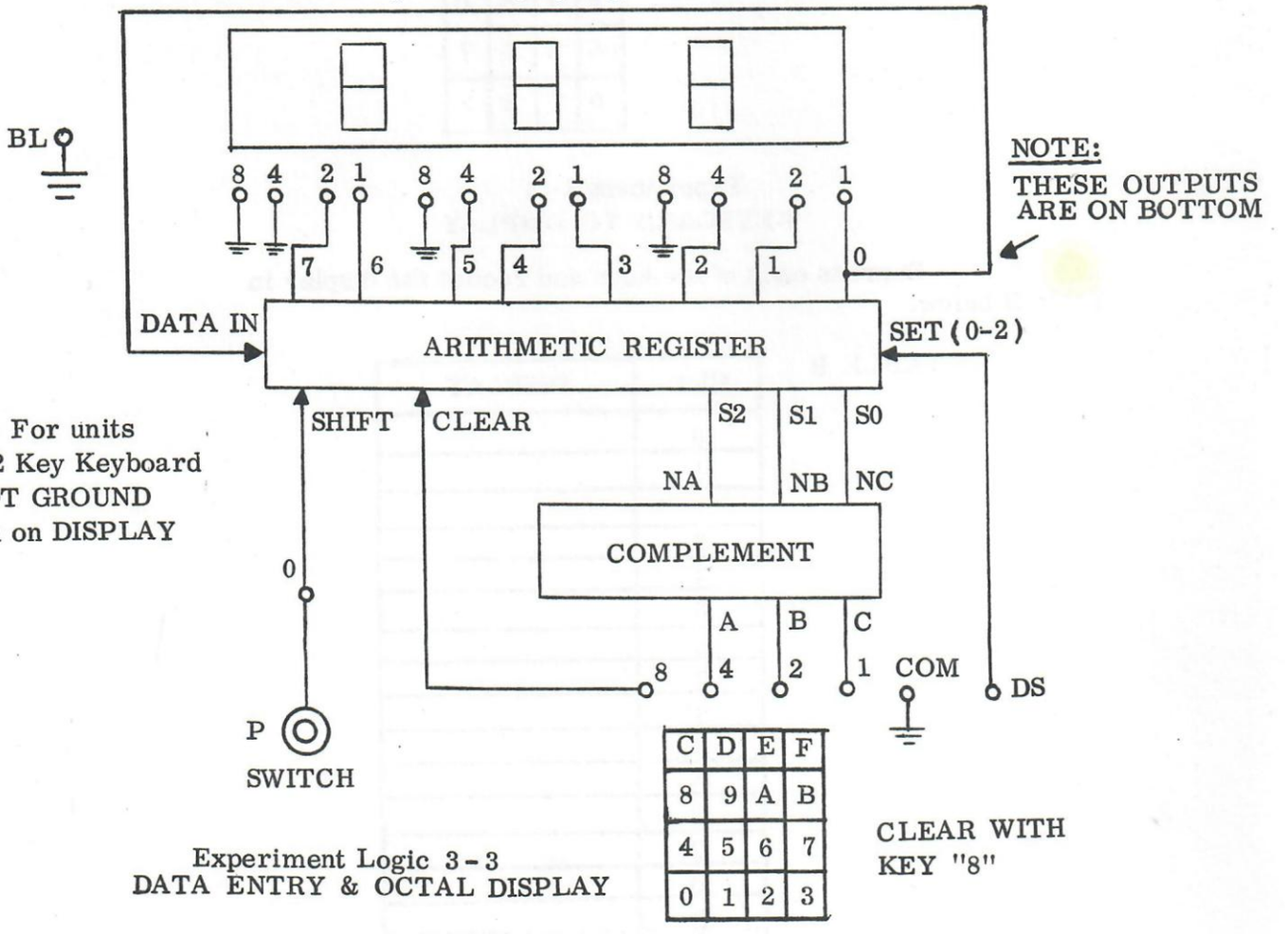
2. The experiment uses a BCD format. How would the keyboard be connected to the display if OCTAL (base 8) numbers were used?

3. If the OCTAL system is used, what is the highest number that can be displayed?

DESCRIPTION (DATA ENTRY & DISPLAY) :

The keyboard is used to "parallel - load" the ARITHMETIC REGISTER with binary numbers and the output of the register is displayed as a 3 digit OCTAL OUTPUT. The ARITHMETIC REGISTER is an eight bit shift register (shifts from bit 7 to bit 0 - left to right). Inputs are provided to bit 7 on the DATA - IN line and shifting is performed when the SHIFT signal switches from a "1" to "0". Bits are available as outputs (0, 1, 2, 3, 4, 5, 6, 7,) and all bits can be set (S0, S1, S2, S3, S4, S5, S6, S7). The bits are cleared to binary "0" when a "0" is applied to the CLEAR line. The bits are set when a "0" is applied to (SET 0 - 2) for bits 0 - 2 and on (SET 3 - 7) for bits 0 - 7.

The Data Entry and Display logic is shown below:



Experiment Logic 3 - 3
DATA ENTRY & OCTAL DISPLAY

When a key is pressed,the DELAYED STROBE enters the data into the first 3 bits (0, 1, 2) of the ARITHMETIC REGISTER. These three bits must first be clear "0" because the strobe (SET 0 - 2) will only set a "1" into the register. When Key 8 is pressed, the Keyboard Output will be grounded. This "8" output is connected to the register CLEAR and the entire register is cleared.

Register Bit 0 is connected to DATA IN so that the entire number entered on the right octal digit (3 bits) can be shifted completely around by pressing the Momentary Switch (P).

The register outputs are connected to the display. It is necessary to ground the display BCD inputs that are not used to provide a binary "0" on these lines. The ground is taken from the GND bus.

PROCEDURE :

Wire the logic shown in Experiment 3 - 3.

Clear the register by pressing the 8 KEY. Enter any number (0 - 7) into the right hand digit by using the keyboard. Shift this number around by pressing P, 5 times. It now appears in the middle digit position. The next numeral is now entered with a KEY and shifted with 5 P shift pulses. The final (least significant right hand) octal digit is now entered via the keyboard. Enter the following three digit numbers into the display and calculate the equivalent decimal number. Complete Table C below:

TABLE C

OCTAL	ENTER	DECIMAL
123		
172		
263		
341		

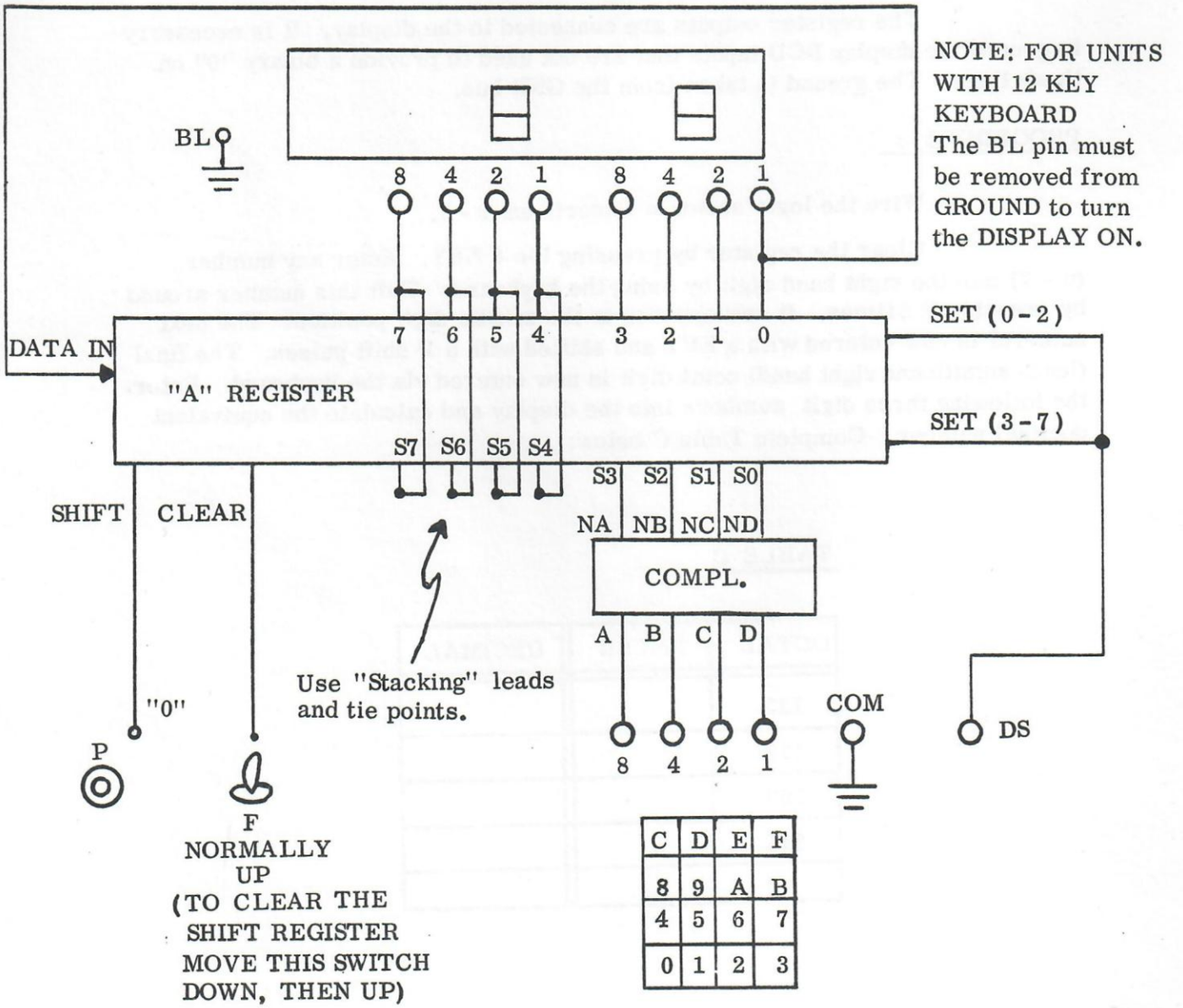
QUESTIONS :

4. Why does 5 shift pulses shift the octal digit one place to the left?

5. What is the maximum numeral that can be stored in the register and displayed on the left hand digit?

DESCRIPTION (BCD coding) :

The 8 bit shift register can store and display two BCD (binary coded decimal) digits as shown in the diagram below:



C	D	E	F
8	9	A	B
4	5	6	7
0	1	2	3

Experiment 3-4
BCD DISPLAY & REGISTER

In the BCD mode, all 4 coded bits (8, 4, 2, 1) from the keyboard are entered into the 4 Right Hand set inputs to the ARITHMETIC shift register and the Shift register is wired into 2 right hand digits of the Display in full BCD format.

Both the SET (0 - 2) and SET (3 - 7) lines are wired together. The outputs 4, 5, 6, 7, are wired to their own set inputs (S4, S5, S6, S7) so that the number that is set is the left hand digit will be strobed in again and thus the number will not change. Because each digit uses a 4 bit BCD code, the register must be shifted with P, 4 times to shift the number. The tie - points and three stacking leads must be used. Clearing is performed using Switch F.

PROCEDURE (BCD coding) :

Wire the logic shown in Experiment 3 - 4. Clear the register by placing Switch F DOWN - then UP and enter a number in the right hand digit by depressing a key. Shift this number into the left - digit by pressing P, 4 times. Enter the following numbers into the register and display. What are the equivalent decimal numbers?

BCD DISPLAY	ENTER	DECIMAL NUMBER
99		
49		
02		
26		

QUESTIONS:

6. If the number 32 is in the shift register, and the register is shifted once; what will the display read?

NOTE:

The Keyboard is composed of an "elastometric" contact material requiring movement of only 0.005 inches. Two "formats" are common as shown below:

7	8	9
4	5	6
1	2	3
+	0	-

CALCULATOR FORMAT

1	2	3
4	5	6
7	8	9
*	0	#

TOUCH TONE FORMAT

C	D	E	F
8	9	A	B
4	5	6	7
0	1	2	3

HEXADECIMAL KEYBOARD

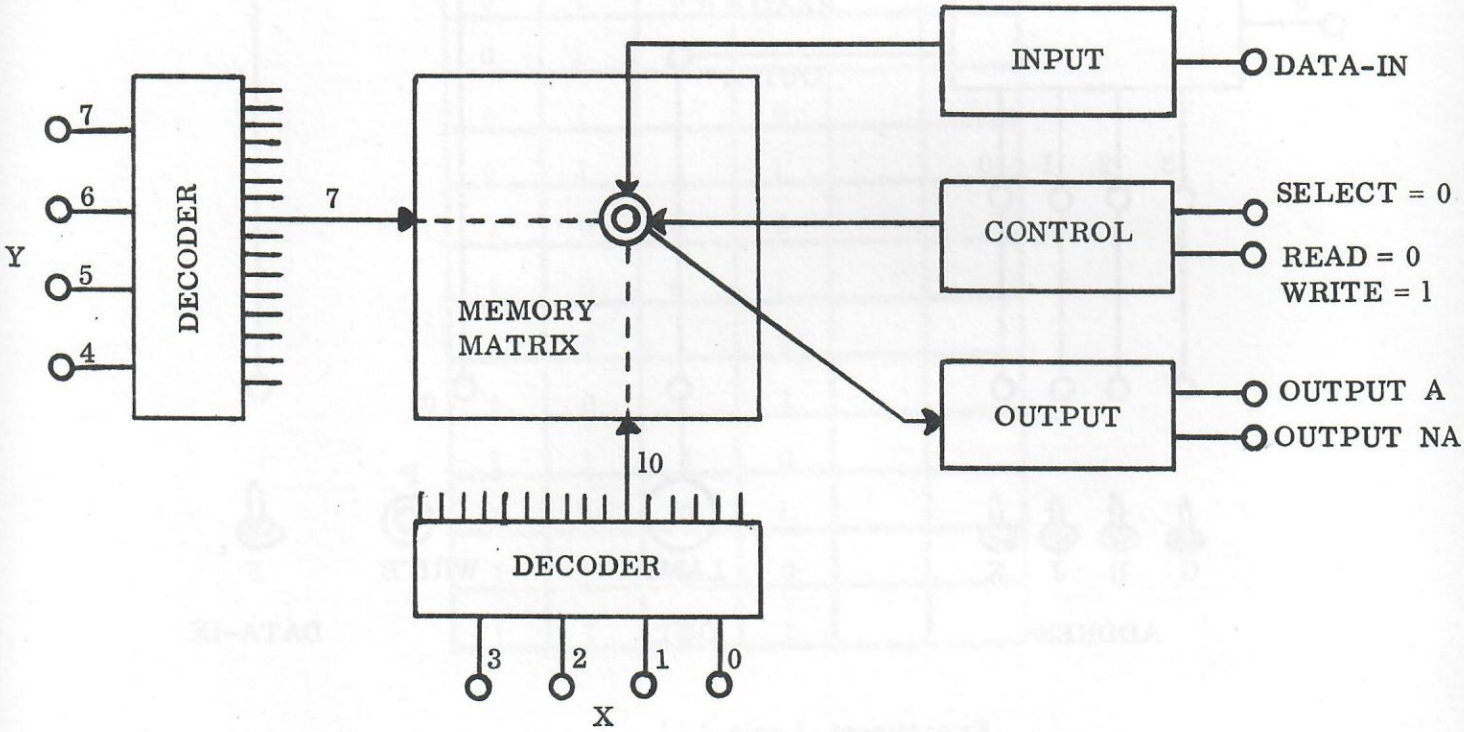
(Used with Microprocessor)

OBJECTIVE:

This experiment demonstrates how a memory location is addressed, how binary data bits (1 or 0) are entered or "read-into" memory, and how data bits are "read-out".

DESCRIPTION:

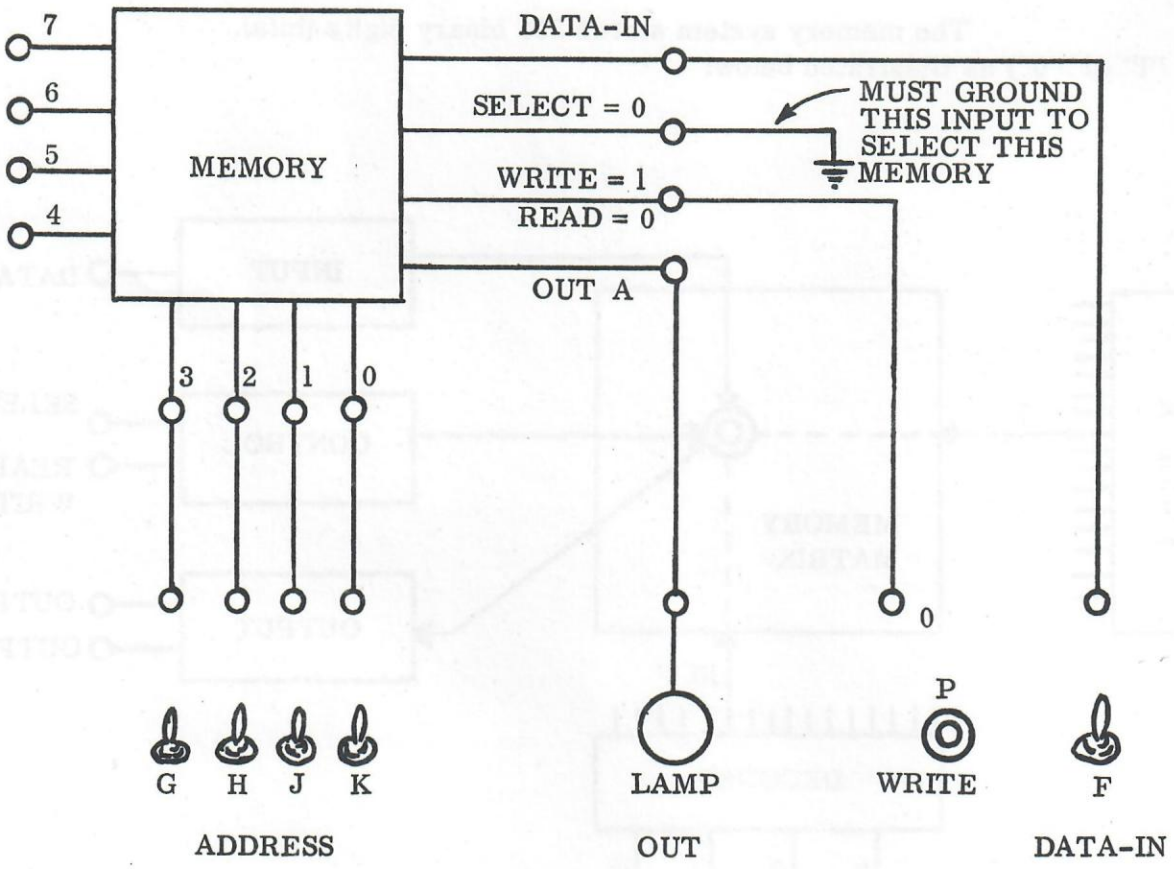
The memory system stores 256 binary digits (bits), (i.e. "1" or "0") as illustrated below:



A specific bit is "addressed" by an eight bit binary number (bits 0 thru 7) via an X decode and a Y decode to select one of 256 bits in a 16 x 16 matrix. A "ground" or "0" applied to the "SELECT = 0" input activates the entire memory "plane" and the bit at the addressed location is presented at "OUTPUT A" and with its complement at "OUTPUT NA". With the "READ/WRITE" input at "ground" or "0", only reading is possible. When "writing" or entering data into memory, the address (bits 0 thru 7) and the "DATA-IN" (1 or 0) must first be applied, and then, a momentary WRITE signal is applied with a "1" to the READ/WRITE input. The READ/WRITE signal must then be returned to "0" to prevent writing in other memory locations when the address is changed.

PROCEDURE (BIT MEMORY):

For simplicity in the logic diagram, the memory is shown as a single block. The Experiment Diagram is shown below:



Experiment Logic 4-1
Random Access; Write-Read

Four address bits are used and set with switches G, H, J, K. (Any four address inputs may be used and others could be used by "grounding" for a "0".) DATA-IN is provided by switch F (1 = UP, 0 = DOWN). The WRITE control is energized using Switch P to apply a "1" for writing into memory. Depressing switch P results in a WRITE = 1 signal applied to the memory. Lamp 10 is used for memory readout A.

Wire the logic, set an address and write bit data in various memory positions entering data in the table below in the SET - IN column. Then READ OUT the data and check the memory. Remember

ADDRESS and DATA - IN must be set before WRITE.

ADDRESS				BIT OF DATA	
G	H	J	K	SET IN	READ OUT
0	0	0	0		
0	0	0	1		
0	0	1	0		
0	0	1	1		
0	1	0	0		
0	1	0	1		
0	1	1	0		
0	1	1	1		
1	0	0	0		
1	0	0	1		
1	0	1	0		
1	0	1	1		
1	1	0	0		
1	1	0	1		
1	1	1	0		
1	1	1	1		

PROCEDURE (WORD MEMORY):

Consider the memory as "storage" for decimal numbers (0 thru 9). Four bits are required for each decimal number (eg. decimal 5 = 0 1 0 1). Addresses J and K are used to decode the four bits as:

J	K	BINARY DIGIT
0	0	1
0	1	2
1	0	4
1	1	8

Switches G and H decode four word locations (0 to 3). Store all of the following numbers in memory, and after all words are stored, read out the data from memory.

WORD ADDRESS	SET J, K (BIT ADDRESS)			
LOCATION Switches (G, H)	DECIMAL WORD INPUT	BINARY WORD (4 BITS)	WORD READ - OUT	CHECK
0 (00)	5	0101		
1 (01)	6	0110		
2 (10)	9	1001		
3 (11)	3	0011		

Repeat the word entry and read - out with any memory words or numbers which you select in the table on the next page.

WORD ADDRESS	USE J, K FOR BIT ADDRESS		
LOCATION (G, H)	WORD INPUT	WORD READ - OUT	CHECK
0			
1			
2			
3			

Change the word in LOCATION 2 only and then read - out the new word.

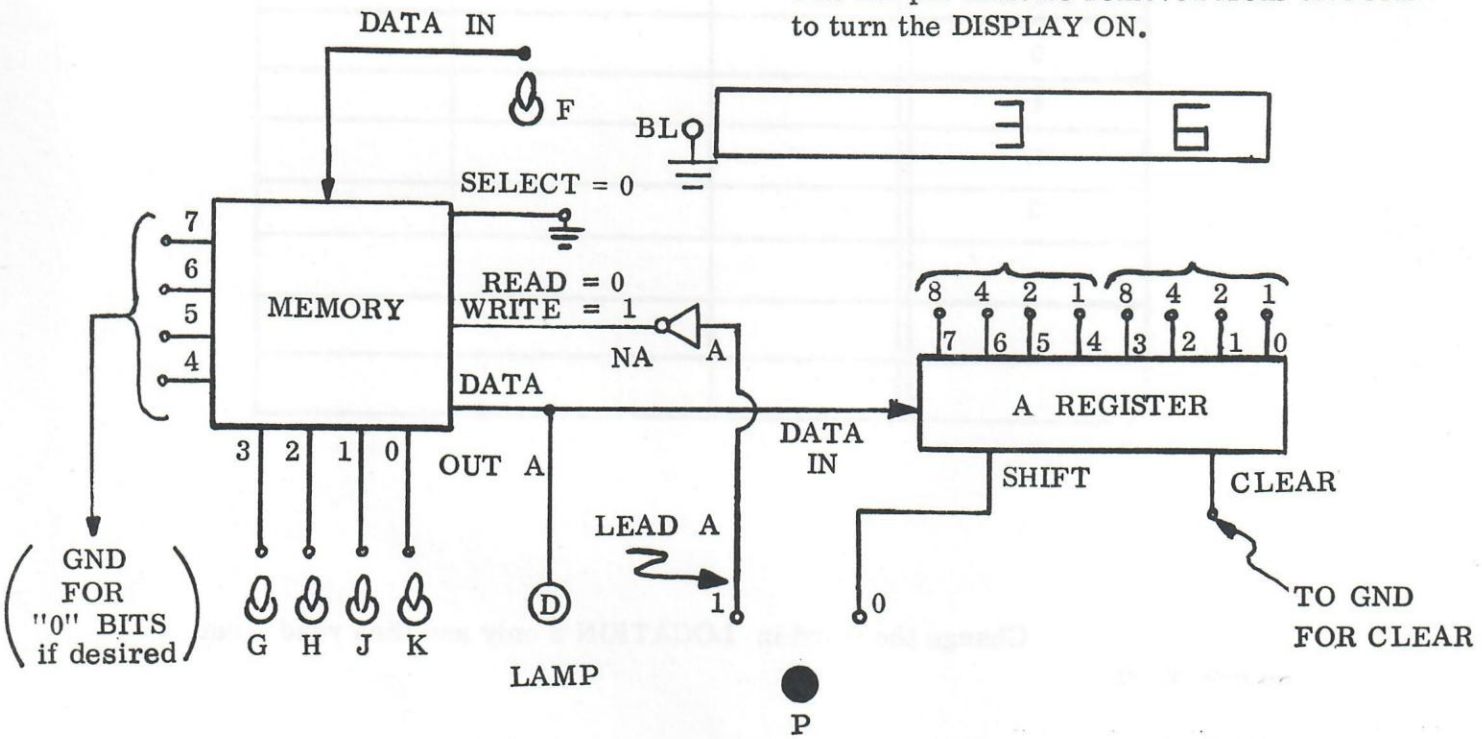
QUESTIONS:

1. How many 4 bit words can be stored in the 256 bit memory?
2. How many 8 bit words can be stored in the memory?
3. How many 6 bit words can be stored in the memory?
4. What is the longest size (bits) word that can be stored in the memory?
5. How many social - security numbers can be stored in the memory?

DESCRIPTION (MEMORY & DISPLAY):

The MEMORY output can be connected to the "A" REGISTER and DISPLAY to provide output data. The ADDRESS is provided by the input switches. The logic is shown below:

NOTE : FOR UNITS WITH 12 KEY KEYBOARD
The BL pin must be removed from GROUND to turn the DISPLAY ON.



Experiment 4-2
MEMORY AND DISPLAY

The Memory Address is provided by Switches G, H, J, K. For data input, "LEAD A" is connected to Switch P "1" via a Complementer NOT element. When Switch P is depressed, a "1" is applied to the Memory WRITE input for writing. The write data bit is applied using Switch F (Up = 1, Down = 0). Additional addresses can be provided by "grounding" memory address inputs 4, 5, 6, 7.

PROCEDURE:

Wire the above logic. Enter data bits via Switch F. Use Switches G, H, J, K for the address. First set Address and Data - In. Depress P to enter the bit into memory. Continue with the next bits and addresses.

Use the numbers and data below:

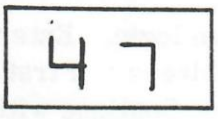
ADDRESS				DATA BIT	
G	H	J	K	Switch F	
0	0	0	0	0	
0	0	0	1	1	
0	0	1	0	0	
0	0	1	1	0	
0	1	0	0	1	
0	1	0	1	1	
0	1	1	0	0	
0	1	1	1	0	
1	0	0	0	1	
1	0	0	1	1	
1	0	1	0	1	
1	0	1	1	0	
1	1	0	0	0	
1	1	0	1	0	
1	1	1	0	1	
1	1	1	1	0	

2 E L F

To read data, disconnect Lead "A". This provides a steady "0" at the memory for "reading".

Switch P is now used to shift data into the A Register. Set the addresses in sequence and read into the A Register. Starting at address 0000, the display should present the memory data after 8 pulses:

0000 to 0111

1000 to 1111 

Enter additional words in memory addresses. Use Switches H, J, K to decode the 8 binary digits of the word (H, J, K, = 000 for bit 0; H, J, K, = 001 for bit 1; etc.). Select various addresses, and enter the data (LEAD "A" connected). Read-out and display the data. Use the table below:

MEMORY ADDRESS					ENTER BCD DATA Switches H, J, K		READ - OUT
7	6	5	4	3	TENS	ONES	
O	O	O	1 (OPEN)	0	0011	0010	32
O	O	O	1	1	0100	0111	47
O	O	O	0 (GND)	0	1001	1000	
1	1	1	0	1			
1	0	0	1	0			

QUESTIONS:

6. Store your telephone number, including area code, in the memory and then read-out the data. How many memory bits are required? How many 8 bit digital words are required?

Enter the data below. Use and OCTAL CODE for the address.

TYPICAL CODING:

TABLE FOR CES PHONE NO.: (516) 293 - 1420

BIT ADDRESS:
SWITCHES (H, J, K);
MEMORY (2, 1, 0)

WORD ADDRESS

ground pins for "0" →

ADDRESS MEMORY INPUTS (6, 5, 4)	DATA
0 0 0	51
0 0 1	62
0 1 0	93
0 1 1	14
1 0 0	20

DESCRIPTION (AUTOMATIC ADDRESS)

In the previous logic circuit (Experiment 4 - 2), the address to the memory was provided by the Switches G, H, J, K. The address was set manually by the operator. In this next experiment logic circuit, the Counters on the Ed-Lab 900 are used to step to the next address when data is entered. The Logic Diagram for Automatic Address Entry is shown on the next page. The P Switch for entering or writing data is connected to the READ/WRITE input via the FULL ADDER input A. It is not necessary to disconnect "Lead A"; the gate in the Full Adder "disconnects" the WRITE signal when Switch H is DOWN.

The logic diagram for the automatic address system is shown below:

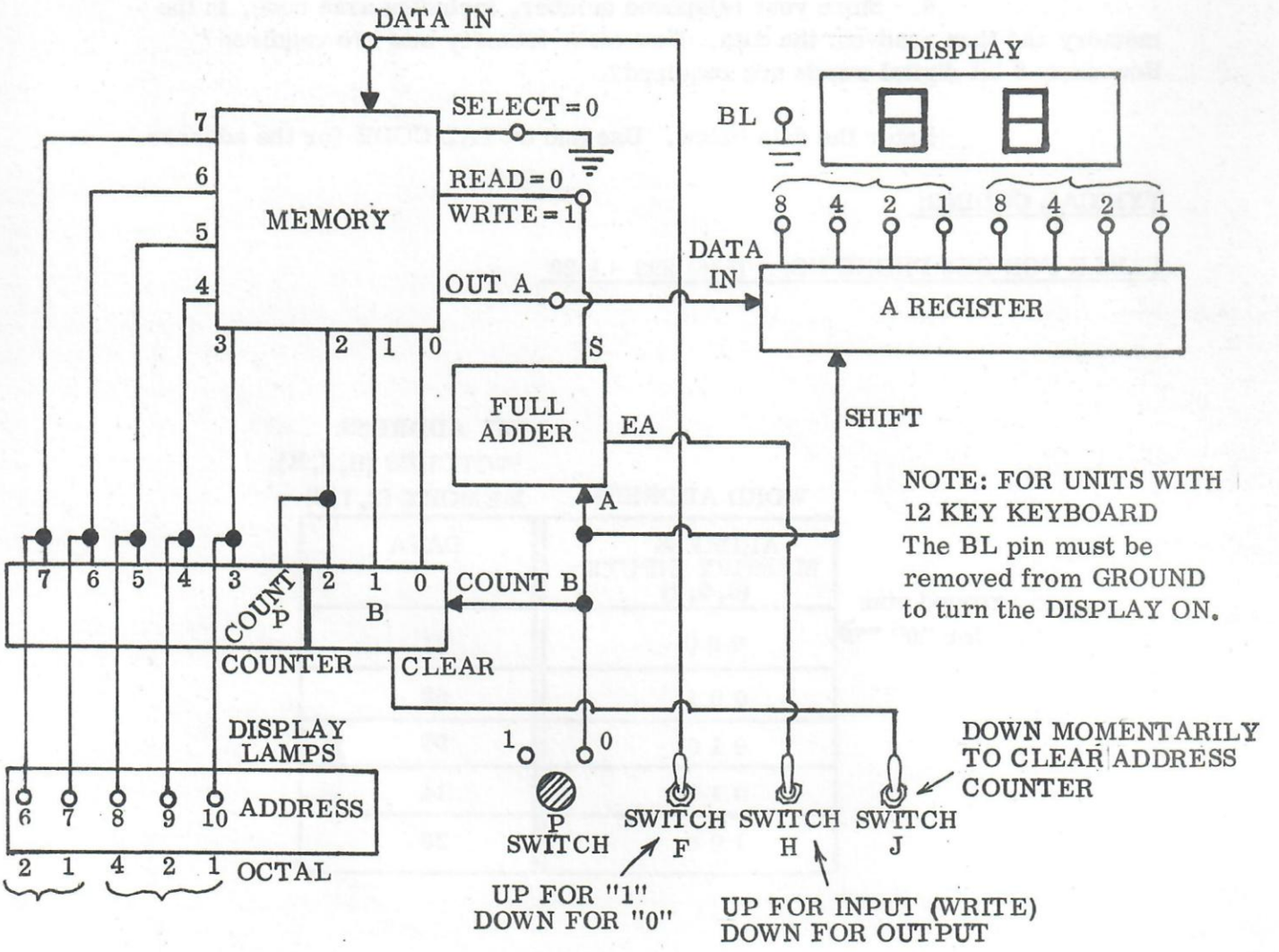


Figure 4-3 AUTOMATIC ADDRESS

Switch J is used to Clear or Reset the address counter (Normally UP, DOWN to Clear).

Switch H is used to enable the WRITE signal. Switch H UP is for WRITE, and Switch H DOWN is for READ.

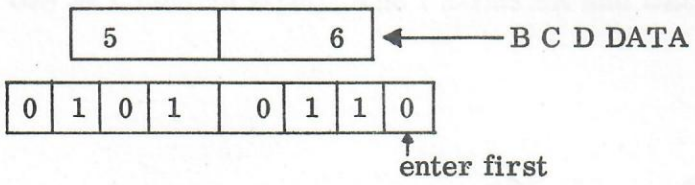
Switch F is used for input data as before (Switch F UP is 1; Switch F DOWN is 0).

Switch P is used to write data as before. It shifts the Register A as before and also steps the Address Counter.

The 5 Output Lights 6, 7, 8, 9, 10 show the Address.

PROCEDURE:

1. Wire the Logic in Figure 4 - 3.
2. Set Switch J DOWN, then UP to Clear the Counter.
3. Set Switch H UP to WRITE data.
4. Enter data using Switch F (1 or 0) and press the P Switch. Continue for 8 steps. Use the data shown below:



5. The data should be in Memory and should also be on the Display. Is the correct data on the Display?_____
6. At the 8th step, the Address Counter should have stepped from 00000 to 00001.
   Is the address correct?_____
7. Continue entering data (with Switch H UP).

   Clear the Address counter whenever desired to start from zero address.

   In order to obtain any address, first clear the counter (Switch J DOWN then UP). Then keep pressing the P Switch until the desired address first appears. When stepping to get to an address, make sure that the WRITE is off; Switch H DOWN.

   Enter the following data:

ADDRESS		DATA	
Binary	Octal	Binary BCD	Decimal
00000	00	01100111	67
00001	01	00110010	32
00010	02	01000001	41
00011	03	01011001	59
00100	04	select your own data	
00101	05		
01011	13		

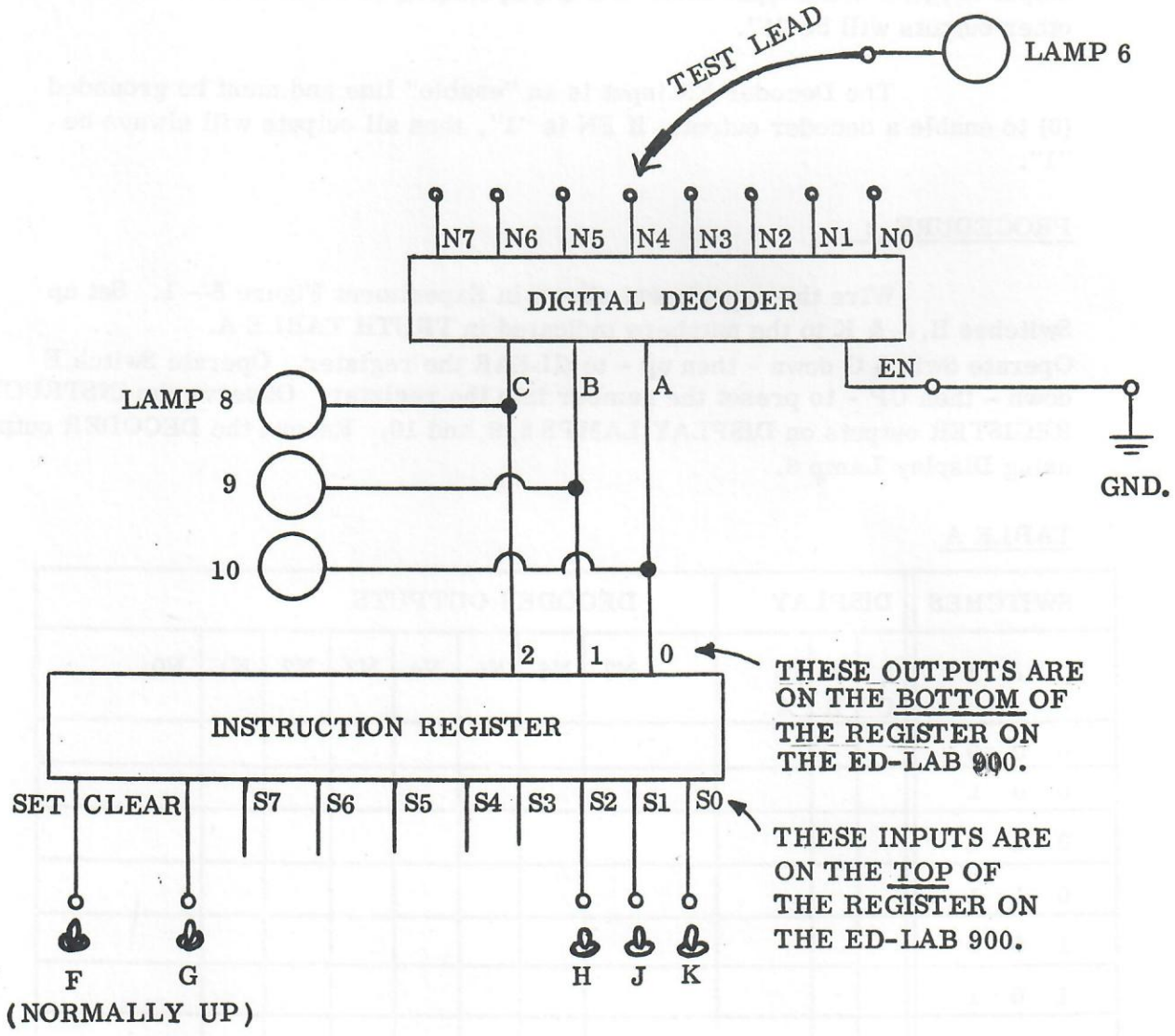
8. With Switch H DOWN (READ), read the data to see if the memory data is correct.

OBJECTIVE :

The objective of this experiment is to demonstrate parallel loading of the INSTRUCTION REGISTER and describe the operation of the DECODER.

DESCRIPTION :

The INSTRUCTION REGISTER is the same as the ARITHMETIC REGISTER with only one SET control input. The REGISTER and DECODE Logic is shown below:



Experiment 5-1
REGISTER AND DECODER

The S inputs to the INSTRUCTION REGISTER are the parallel load inputs. These will only load a "1" into the register. The register must be "cleared" before attempting to parallel load a number. The CLEAR signal is a ground ("0") which clears all shift register outputs to binary zero. The desired number is then set up on the preset (parallel load) switches, H, J, K, and the SET input is momentarily grounded. The shift register outputs then correspond to input switch settings. (An open input is the same as a "1" input).

The DECODER is a 1 out of 8 decoder. A 3 bit binary number (0 - 7) is applied to the A, B, & C input lines. The decoded binary input results in a "low" output. Outputs are normally "1" and only the selected or decoded output is "0". If the input number is (A, B, C=110), then N3 will be "0" and all the other outputs will be "1".

The Decoder EN input is an "enable" line and must be grounded (0) to enable a decoder output. If EN is "1", then all outputs will always be "1".

PROCEDURE :

Wire the experiment shown in Experiment Figure 5 - 1. Set up Switches H, J, & K to the numbers indicated in TRUTH TABLE A. Operate Switch G down - then up - to CLEAR the register. Operate Switch F down - then UP - to preset the number into the register. Observe the INSTRUCTION REGISTER outputs on DISPLAY LAMPS 8, 9, and 10. Record the DECODER outputs using Display Lamp 6.

TABLE A

SWITCHES	DISPLAY			DECODER OUTPUTS							
H J K	8 C	9 B	10 A	N7	N6	N5	N4	N3	N2	N1	N0
0 0 0											
0 0 1											
0 1 0											
0 1 1											
1 0 0											
1 0 1											
1 1 0											
1 1 1											

Leave switches F and G DOWN. Observe that the output (Lamps 8, 9, 10) follow the input Switches H, J, K.

Check the outputs with line EN not connected to ground. Comment on the results.

QUESTIONS :

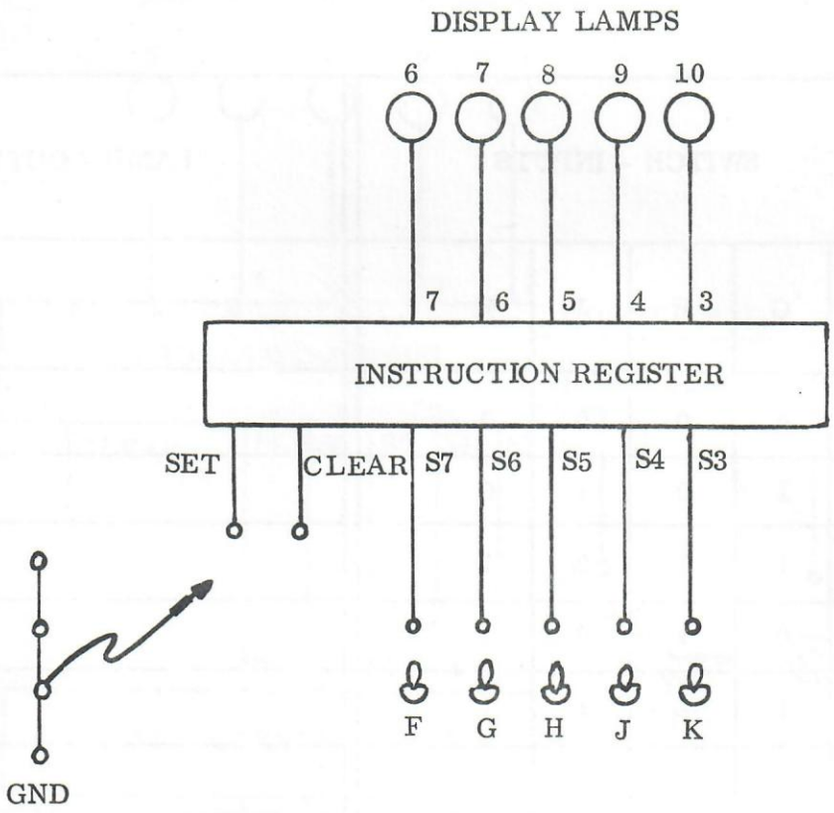
1. How many outputs are there for a 4 input DECODER?

OBJECTIVE :

The object of this experiment is to introduce the INSTRUCTION REGISTER (I) and the DIGITAL MULTIPLEXER (M). The MULTIPLEXER is then utilized to select either the PROGRAM COUNTER outputs or the INSTRUCTION REGISTER outputs.

DESCRIPTION (INSTRUCTION REGISTER) :

The INSTRUCTION REGISTER is similar to the ARITHMETIC REGISTER. The only difference is that there is only one SET input line. The Logic Diagram is shown below:



Experiment 6 - 1

PRESET INSTRUCTION REGISTER

Switches F, G, H, J and K are connected to the preset inputs S3 thru S7. Note that the panel has these inputs on the top of REGISTER.

PROCEDURE :

Wire the circuit shown in Experiment 6 - 1. Enter the Input Switches in various conditions as shown in Table A and record the outputs in Table A. Clear the register by "touching" the CLEAR line to ground. Then set the number by touching the SET line to ground. Enter the data below:

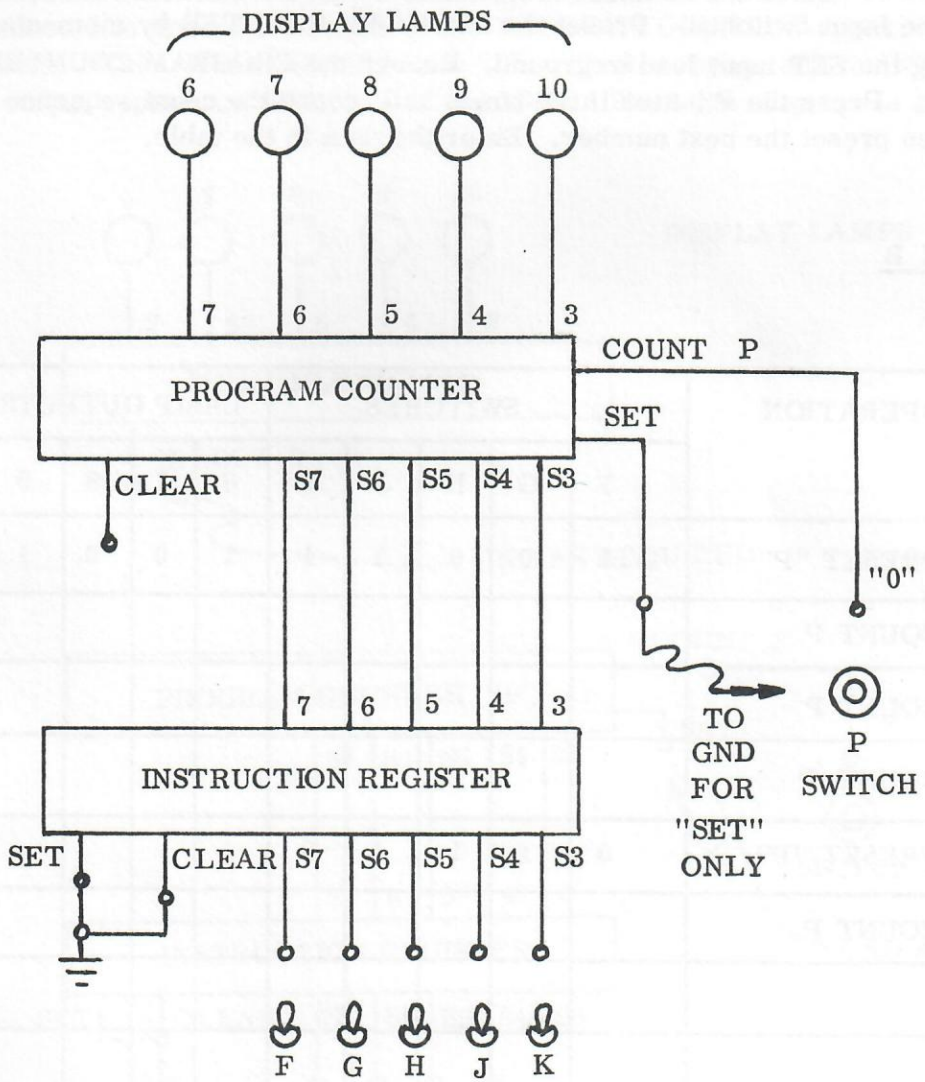
TABLE A

SWITCH - INPUTS					LAMP - OUTPUTS
F	G	H	J	K	6 7 8 9 10
0	0	0	0	0	
0	1	0	1	0	
1	1	1	0	1	
1	0	1	0	1	
1	1	1	1	1	

Connect the SET and CLEAR lines both to ground. Vary the Input Switches. Observe that the SET "1" input predominates and the output (lamps) follows the input (switches).

DESCRIPTION (COUNTER PRESET - DATA TRANSFER):

In the above and previous experiments, the Registers were preset with binary numbers determined by Switch Inputs. The diagram below shows the PROGRAM COUNTER preset inputs S3, S4, S5, S6, S7 connected to the INSTRUCTION REGISTER outputs.



Experiment 6 - 2
PRESET DATA TRANSFER

When the SET input on the PROGRAM COUNTER is momentarily grounded, the number in the INSTRUCTION REGISTER is transfered into the PROGRAM COUNTER. It is not necessary to clear the Program Counter before setting. COUNT pulses into the counter will cause it to count from this pre-set number. The number set with the Input Switches is automatically loaded into the INSTRUCTION REGISTER with the SET and CLEAR lines both grounded. When the ground Wire is connected to the SET line of the PROGRAM COUNTER, the number in the INSTRUCTION REGISTER is TRANSFERED to the program counter. Push button P is used to clock the counter and count from the preset number.

PROCEDURE (TRANSFER) :

Enter the numbers from Table B into the INSTRUCTION REGISTER using the Input Switches. Preset the PROGRAM COUNTER by momentaily touching the SET input lead to ground. Record the PROGRAM COUNTER output number. Press the P button three times and record the count sequence in Table B. Then preset the next number. Enter the data in the table.

TABLE B

OPERATION	SWITCHES					LAMP OUTPUTS				
	F	G	H	J	K	6	7	8	9	10
PRESET "P"	1	0	0	1	1	1	0	0	1	1
COUNT P										
COUNT P										
COUNT P										
PRESET "P"	0	1	1	1	0					
COUNT P										

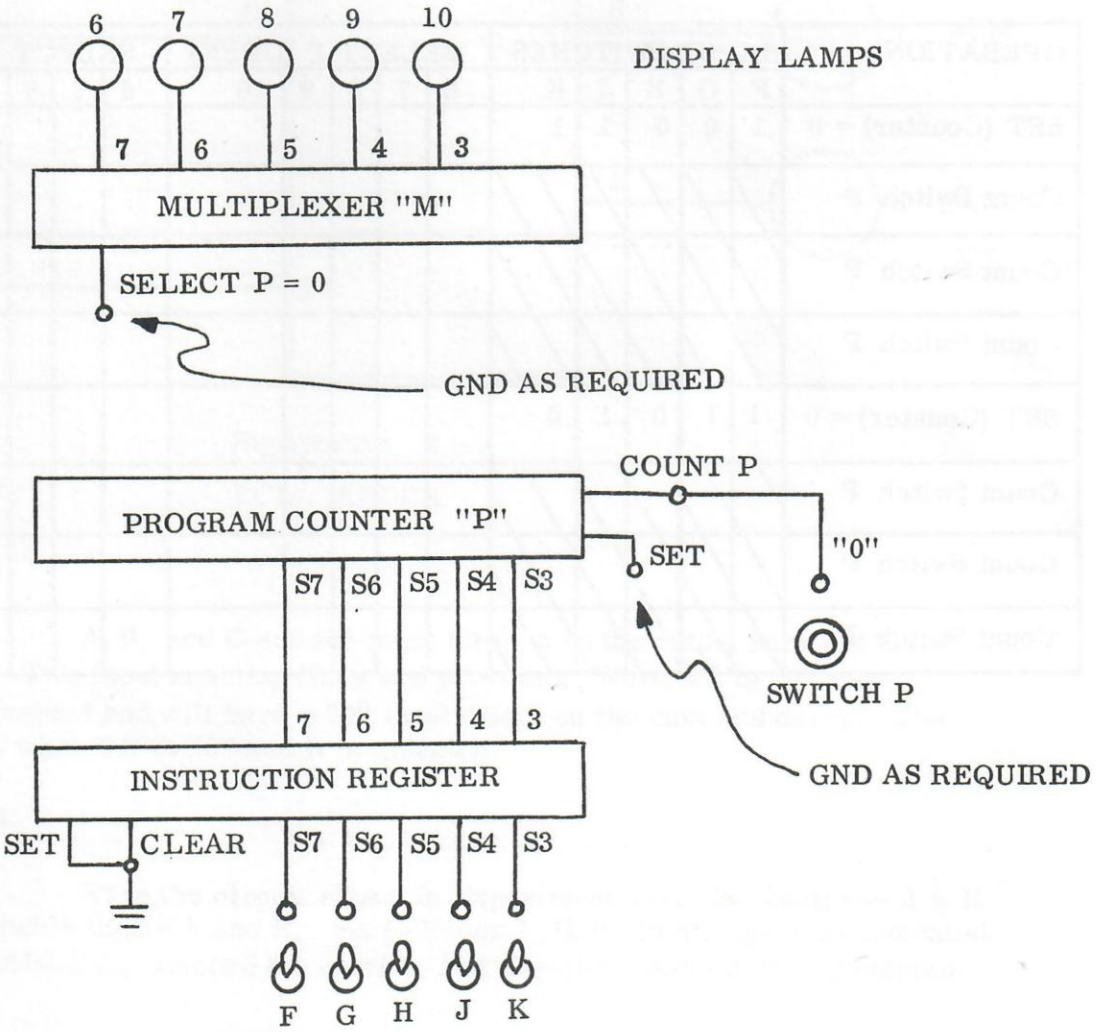
QUESTIONS :

1. Will the PROGRAM COUNTER count if the SET line is left connected to ground?

2. What logic circuit would be used to provide an 8 bit program counter with 8 bit Instruction Register and 8 bits of TRANSFER?

DESCRIPTION (MULTIPLEXER) :

The MULTIPLEXER or "MUX" is internally connected to the outputs of both the INSTRUCTION REGISTER and the PROGRAM COUNTER (COUNTERS P and B). Either one or the other of these outputs is applied to the output of the MUX depending on the control signal into the SELECT input of the MUX. When SELECT P = 0 is connected to ground,the output of the PROGRAM COUNTER appears at the output of the MUX. When SELECT is at "1" or "not - connected" the INSTRUCTION REGISTER output appears at the output of the MUX. The logic diagram is shown below:



Experiment 6 - 3
MULTIPLEXER

The Lamp Display output is now taken from the MUX. When SELECT P=0 on the MUX is grounded, the counter output is displayed. When P is a "1" or floating, the Instruction Register output is displayed. When SET on the counter is grounded, the Instruction Register output is Transfered to the counter.

PROCEDURE (MUX) :

Set the Switches as indicated in Table C and "ground - and - lift" the MUX control P to monitor the register and counter outputs for each step in the operation. Enter the data in the table below:

TABLE C

OPERATION	INPUT SWITCHES					SELECT P (OPEN)					SELECT P = (GROUND)				
	F	G	H	J	K	6	7	8	9	10	6	7	8	9	10
SET (Counter) = 0	1	0	0	1	1										
Count Switch P															
Count Switch P															
Count Switch P															
SET (Counter) = 0	1	1	0	1	0										
Count Switch P															
Count Switch P															
Count Switch P															

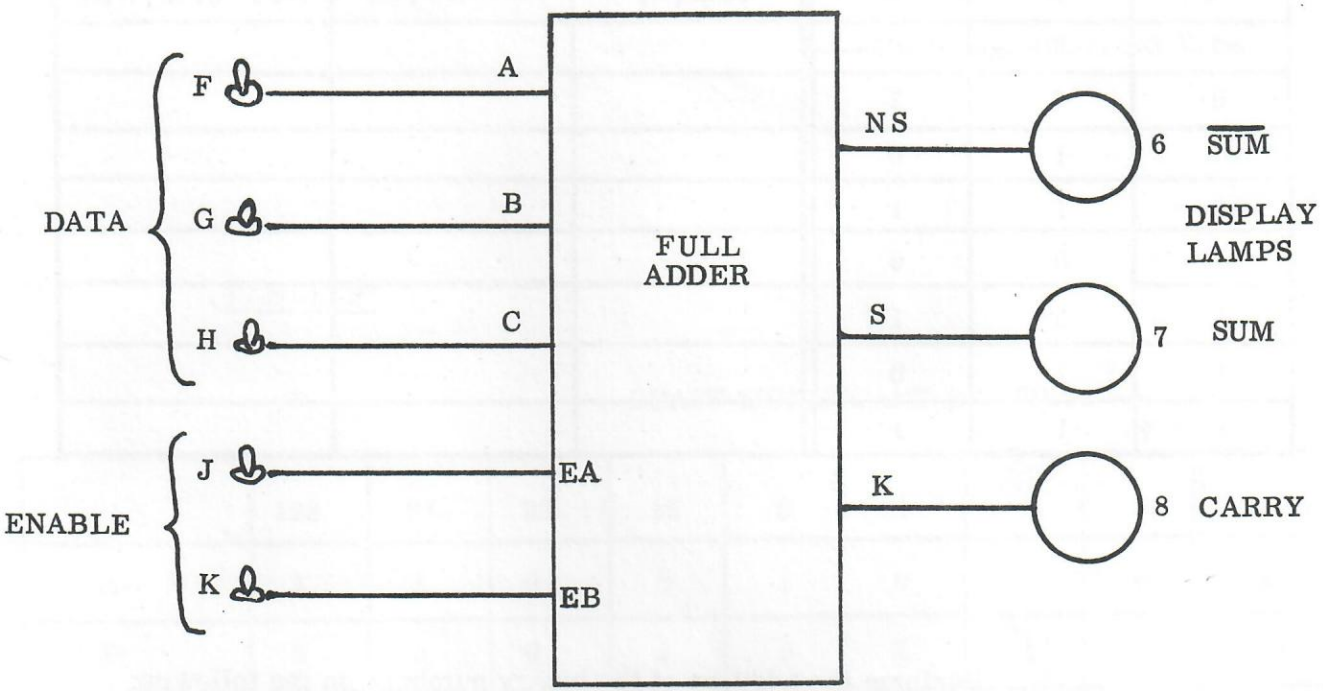
ADDER AND OCTAL ARITHMETIC EXPERIMENT NO. 7

OBJECTIVE:

This experiment demonstrates the operation of the arithmetic ADDER to add and subtract binary numbers.

DESCRIPTION:

The ADDER is a Full Adder circuit and is shown below:



Experiment 7 - 1

FULL ADDER

A, B, and C are the input bits. S is the output sum and K is the CARRY. Two input enabling lines are provided. When AE is "0" the A input is disabled and will have a "0" input effect on the sum and carry. The same is true when BE is "0" and B is disabled.

PROCEDURE:

Wire the circuit shown in Experiment 7-1. Set Switches J & K both UP to enable inputs A and B. Set Switches F, G, H, to all inputs as indicated in TRUTH TABLE A. Record the Sum(S); Not Sum(NS), and Carry (K) outputs.

TRUTH TABLE A

FULL ADDER

A	B	C	SUM (S)	CARRY (K)	NOT SUM (NS)
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

Perform the addition of the binary numbers on the following page. This is done by setting the Least Significant bit of numbers A and B into Switches F & G. The carry input Switch H should initially be "0".

Record the sum (S) and carry (K) output and set the next bits with Switches F & G and the "last carry output" with Switch H. Repeat this process for all the bits. Record the results in Table B.

ADD NUMBERS:

A 1 1 0 0 1 0 1 0 (SWITCH F)
+B 0 0 0 1 0 1 1 1 (SWITCH G)

Least Significant Bit First

TABLE B

LEAST SIGNIFICANT BIT FIRST

	128	64	32	16	8	4	2	1	DECIMAL
A	1	1	0	0	1	0	1	0	202
+ B	0	0	0	1	0	1	1	1	
C								0	Not Used
(K) CARRY									Not Used
(S) SUM									

Disable input B by connecting EB to "0" and connect the input C to "0". Set various inputs of A and B and observe that the output (S) follows A. Enter data in the table below.

TABLE C

EB = 0 C = 0

A	B	S
0	0	
0	1	
1	0	
1	1	

Disable input A by connecting EA to "0". Observe that the output (S) follows the input B. Set Switch C to "0".

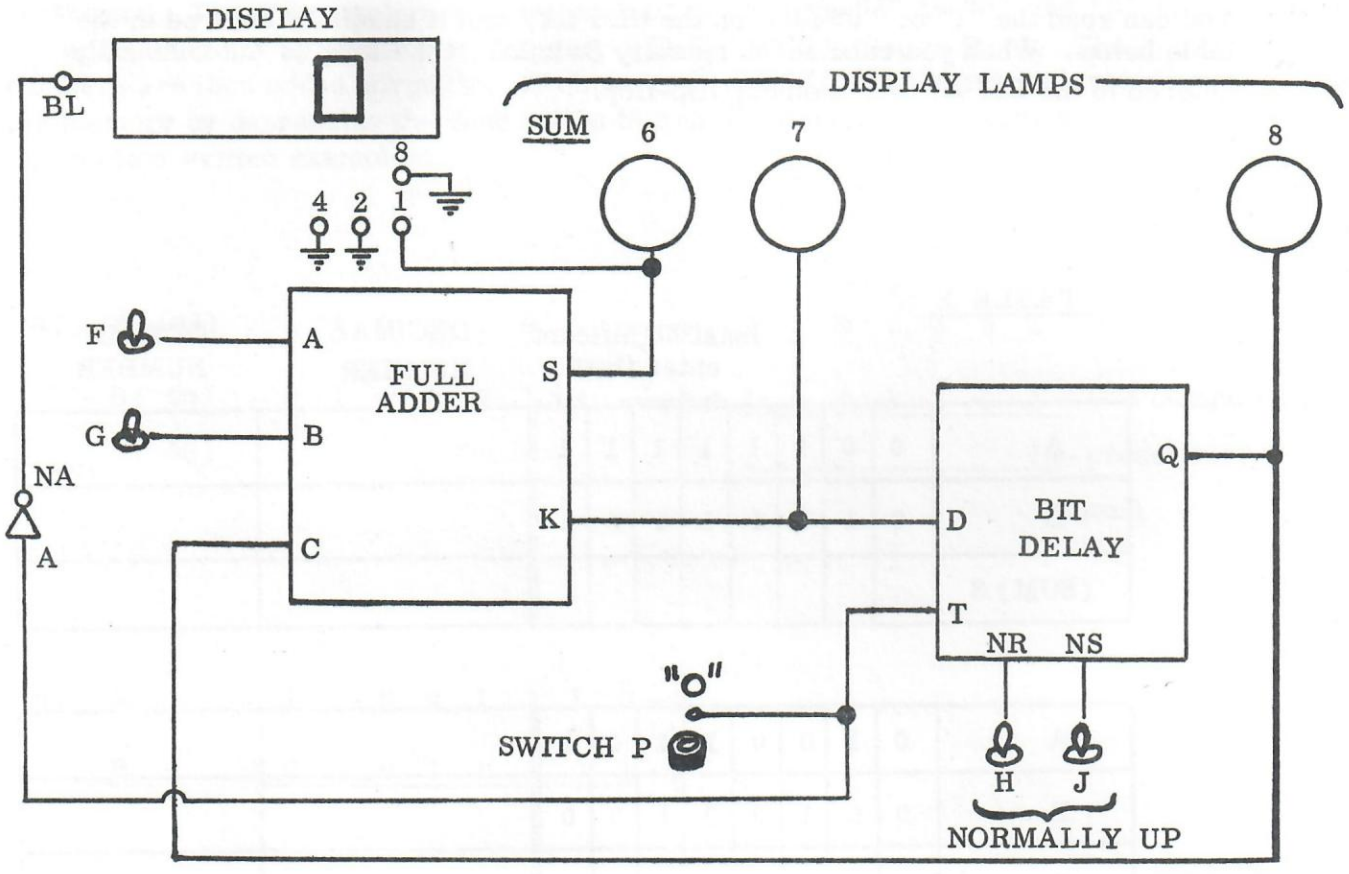
TABLE D

EA = 0 C = 0

A	B	S
0	0	
0	1	
1	0	
1	1	

DESCRIPTION (Adder with Memory) :

In this portion of the experiment, a BIT DELAY (flip-flop) memory is used at the carry K output and C input to automatically provide the carry bit from the previous bit addition to the C input. The logic is shown below:



Experiment 7 - 2
ADDER WITH MEMORY

NOTE: FOR UNITS WITH 12 KEY KEYBOARD delete inverter A. Connect directly from "P" Switch to BL Pin.

The Momentary Switch P is used to enter the carry data (K) into the BIT DELAY flip-flop. Switches H & J are used to initialize or preset the carry. When adding, the carry memory is cleared to Q = 0 using Switch H. When subtracting one number from another, the number to be subtracted is complemented (one's complement) and the carry is initialized at Q = 1 using Switch J. The normal addition is then performed. The display is used to present a "1" or "0" in normal decimal numerals. The P Switch "unblanks" the display when the switch is depresssed.

PROCEDURE :

Wire the circuit shown in Experiment 7-2. Reset the carry memory (Q = 0) (Lamp 8 out) using Switch H (down, then up). Switch J should be UP. Set in the bits for the binary number A & B with the least significant bit first. After A & B are set-up, depress Momentary Switch P and hold it in. You can read the "1" or "0" SUM on the DISPLAY and it should be entered in the table below. When you release Momentary Switch P, the Carry is Automatically entered in the BIT DELAY memory flip-flop.

TABLE E

									least significant enter first ↓ DECIMAL NUMBER	OCTAL NUMBER
A	0	0	1	1	1	1	1	1		
+ B	0	1	0	1	1	1	0	1		
(SUM) S										

A	0	1	0	0	1	1	0	1		
+ B	0	0	1	0	0	1	0	0		
S										

A	0	0	0	0	0	1	1	1		
+ B	0	0	1	1	0	1	1	0		
S										

Write the numbers A and B and the Sum S in decimal form and check the addition.

PROCEDURE (Subtraction) :

To perform subtraction we add the two's complement of the subtrahend. The 2's complemented is obtained by changing "1" to "0" and "0" to "1" and adding "1" by presetting the carry BIT DELAY flip-flop to Q = 1. The numbers are then added normally. Each carry output (K) must be entered into the memory by depressing the push button Switch P. Perform the following subtraction written examples:

(a) A(109) 1 1 0 1 1 0 1 → 1 1 0 1 1 0 1

- B(50) - 0 1 1 0 0 1 0 → + 1 0 0 1 1 0 1 (1's complement)

(59) 1 ← preset carry

← sum

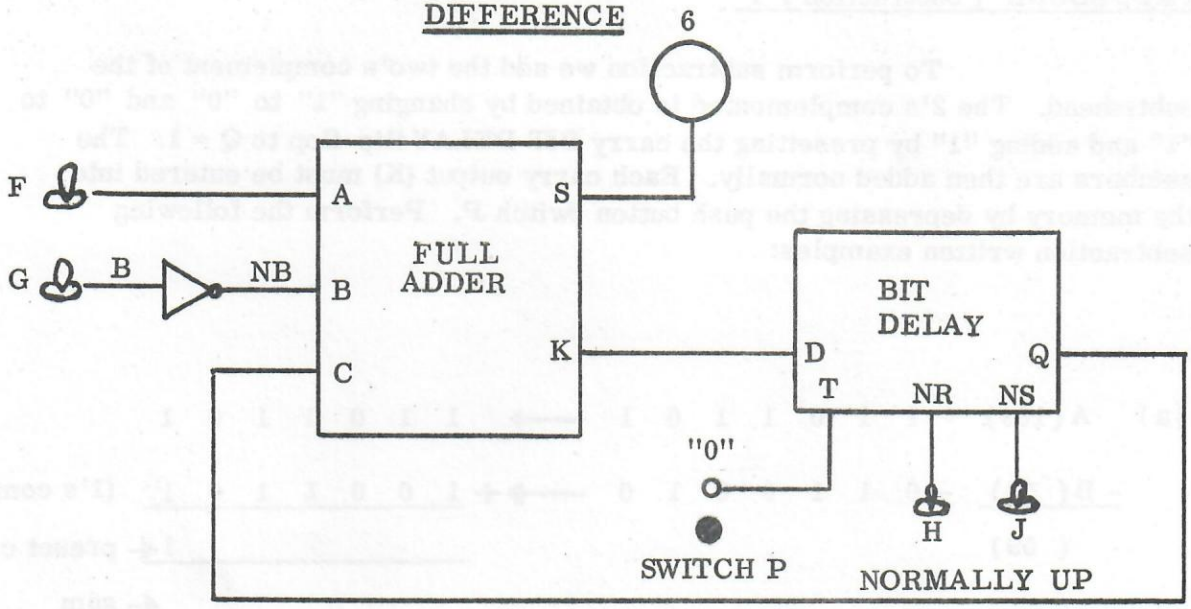
(b) A 1 1 0 0 1 0 1 0 →

- B - 0 0 0 1 1 1 0 1 →

(c) A 1 0 0 1 1 0 0 0 →

- B 0 0 1 1 0 0 0 0 →

Insert an Inverter in the B input to obtain the 1's complement as shown below.



Experiment 7-3
SUBTRACTION WITH MEMORY

Start with Switches H & J up. For Subtraction, Preset a CARRY with Switch J Down then Up. Run the problem as before starting with the least significant bit.

Enter the data in the table below. The most significant digit specifies the sign of the number. A "1" is negative. To find the positive number, find the 1's complement and add 1.

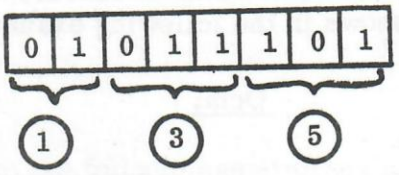
TABLE F

least significant bit

										check decimal	octal
	A	0	1	0	1	1	0	1	0		
MINUS	B	0	1	0	1	0	0	1	1		
	DIFF.										

										check decimal	octal
	A	0	1	1	0	0	1	0	1		
MINUS	B	0	0	1	1	1	0	0	1		
	DIFF.										

Write the answers in OCTAL. A typical conversion is:



QUESTIONS :

1. Convert the following binary numbers to octal:

(a) 00010111
(b) 11010010
(c) 11101001

2. Determine the sign and magnitude of the above numbers.

DESCRIPTION (OCTAL SUBTRACTION) :

In order to perform subtraction and represent negative numbers, the 2's complement binary number system is used. When computing in OCTAL, the 8's complement number system is used.

The following example is given:

OCTAL 023
8's compl. 355

adds to 8
adds to 7
adds to 3 (because only 2 bits are in left hand digit in the 8 bit word)

Note when we add :

```
  023
+ 355    (complement of 023)
  000    answer is zero.
```

Questions :

3. Find the 8's complement of the octal numbers:

(a) 024
(b) 034
(c) 257
(d) 347

To subtract, the subtrahend is converted to its 8's complement and addition is performed as shown in the following example:

Decimal	Octal		
19	023	→	023
- 38	- 046 → 8's compl.	→	+ 332
-19			355 answer is negative number

Converting 355 to sign plus magnitude:

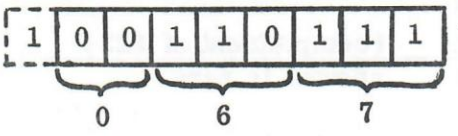
355
8's compl. → 023 result is -19 (decimal)

A left hand digit of 2 or 3 indicates the left hand bit of the binary number is a "1" and therefore the number is NEGATIVE. The left hand digit used in the computer and in our problems has a maximum value of 3 because there are only 2 bits in the left hand digit.

If there is an overflow in the additions, the left hand digit would register 4, 5, 6, etc. Because there are no further bits, the most significant digit is lost. Thus, adding:

```
  125
+ 342
  067
```

The normal sum is 467 but the 4 bit is lost in the binary "overflow" as shown below;



The following exercises provide practice in the addition and subtraction of octal numbers.

QUESTIONS :

4. Perform the following operations and check with the decimal equivalents:

	OCTAL	DECIMAL
(a)	035 +027	
(b)	125 +032	
(c)	(Answer will be wrong because of overflow.) 077 +136	
(d)	077 +236	
(e)	372 +157	
(f)	072 -157	
(g)	157 -072	
(h)	234 -321	
(i)	234 -121	

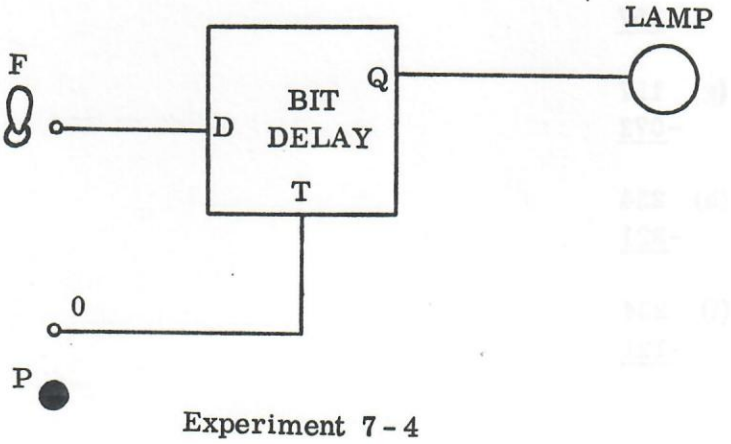
NOTE CAREFULLY:

J - K FLIP - FLOP / MASTER SLAVE

The J - K FLIP - FLOPS employed in the Ed - Lab 700 Computer Systems Lab and on the BIT DELAY in the Ed - Lab 900 Memory and Display Lab is the most commonly used type called the "master - slave". The J and K inputs must not be allowed to change states when the clock or trigger input is HIGH (1) or the output will "complement" on the negative going voltage transition of the trigger.

The normal operation is to keep the trigger input low (0) while changing J and K, and then to pulse or trigger to "1" and "0" to transfer information.

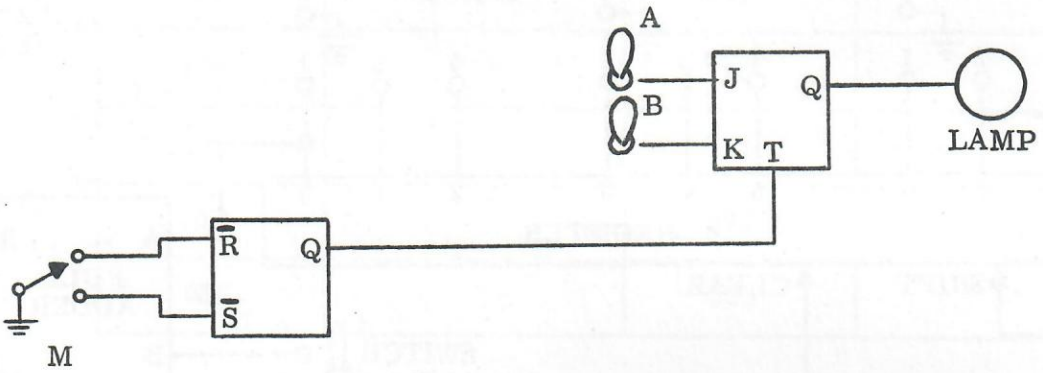
This operation can be observed on the Ed - Lab 900 Bit Delay as shown below, by varying D when the Switch P is depressed.



Experiment 7 - 4

BIT DELAY MASTER SLAVE

The operation can be observed on the Ed - Lab 700 using Switch M and a flip - flop for a pulse as shown below:



Experiment 7 - 5

MASTER - SLAVE TRIGGERRING

Comment on the results._____

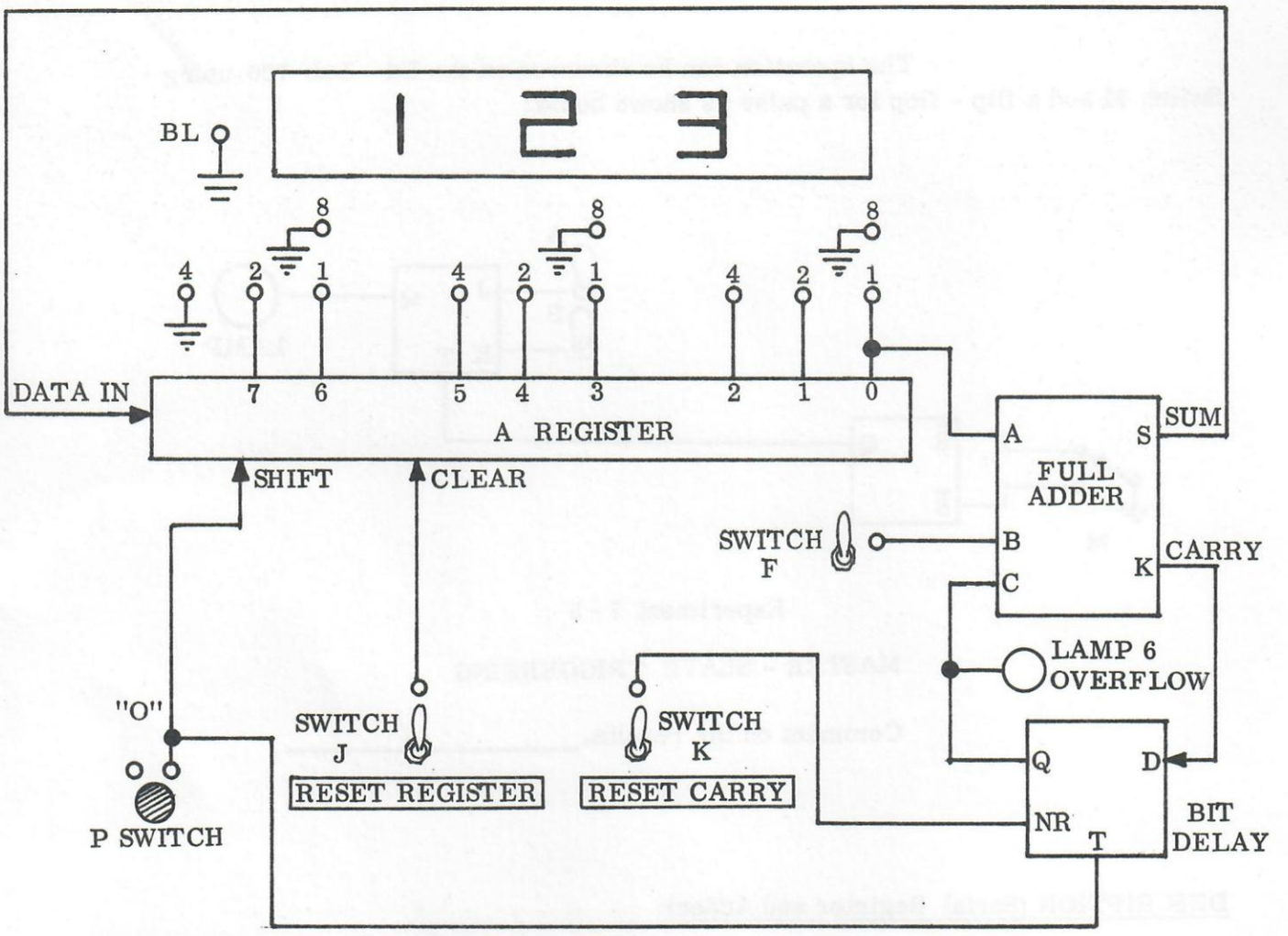
DESCRIPTION (Serial Register and Adder):

The Full Adder can be used with a Shift Register to store or Accumulate the results of the Addition. The logic diagram is shown in the figure for Experiment 7 - 6 on the following page. The "A Register" (A is for ACCUMULATOR) is connected to the Adder input "A" and the Sum "S" is returned to the DATA-IN terminal on the A Register. The Full Adder and Carry Flip-Flop or Bit Delay is connected as before with the Carry Shift or Trigger connected to the Shift Register SHIFT input.

Switch J Clears the A Register (Normally UP;; Clear DOWN then UP)

Switch K Clears the Carry memory (Normally UP; Clear DOWN then UP)

Switch F is the Input Data (1 or 0).



NOTE: FOR UNITS WITH 12 KEY KEYBOARD
The BL pin must be removed from GROUND
to turn the DISPLAY ON.

EXPERIMENT 7-6 SERIAL REGISTER AND ADDER

PROCEDURE(Serial Register and Adder):

1. Wire the Logic given in Experiment 7 - 6.
2. Use Switch J to Clear the A Register (DOWN then UP).
3. Use Switch K to clear the Carry Memory (DOWN then UP).
4. Set Switch F for the input bit and then depress the P Switch. Continue for 8 bits of Data to ADD numbers. Do not Clear the A register which accumulates the Sum. It is not necessary to clear the Carry Memory unless there is an Overflow, indicated by an ON at Lamp 6. Add the following numbers taken from Questions 4(a) and 4(b) and 4(d).

	NUMBER	BINARY (ENTER)	SUM (READ)	CHECK
	035	00011101	035	
	027	00010111	064	
RESET "A" (Switch J) →	125		125	
	032		157	
RESET "A" (Switch J) →	077		077	
	236		335	

5. Repeat the Addition test by Accumulating added results as given in the table below:

NUMBER	BINARY (Enter)	SUM	CHECK
035			
027			
125			
032			
077			
035			
236		(OVERFLOW)	

Note that there is an Overflow in the last addition resulting in a "1" in the Carry Memory and an incorrect sum.

DESCRIPTION: (Instructions and Arithmetic Unit)

This experiment combines the Arithmetic Unit or Adder and Accumulator with an Instruction Code for Adding, Subtraction, Loading (which is a Clear and Add), and Storing (the Store is an output instruction that ignores the input B on the Adder).

Subtraction was described previously as performed by Inverting (One's Complement) the B input, Adding, and Adding a "1". A typical example is 7 minus 3 which results in 4 and is performed in binary as shown below:

A	7	0111 ⟶	same	0111
-B	3	0011 ⟶	invert	1100
		Add "1"		0001
		Sum		0100 Difference is 4
		ignore overflow		

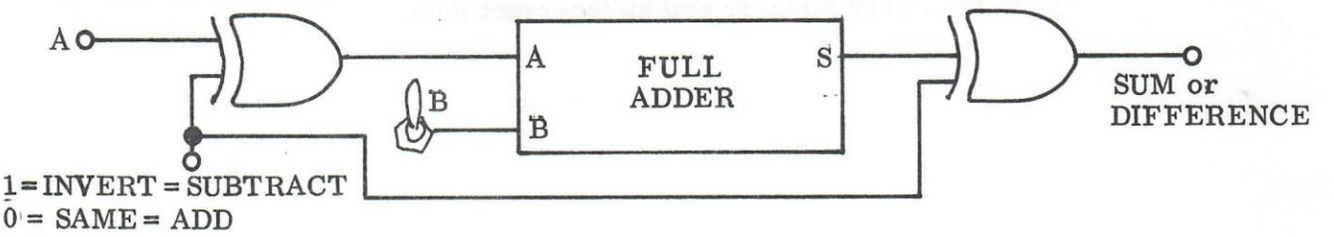
Another way of subtracting is to Invert (One's Complement) the A Input; then Add; then Invert the Sum to obtain the Real answer or Difference. The same example is shown below:

A	7	0111 ⟶	invert	1000
-B	3	0011 ⟶	same	0011
			sum	1011
		⟶	invert	0100 Difference is +4

Another example is:

A	6	0110 ⟶	invert	1001
-B	3	0011 ⟶	same	0011
			sum	1100
		ignore overflow ⟵		
		⟶	invert	0011 Difference is +3

The Exclusive-Or performs the function of the Inversion or One's Complement. A "0" input to the EX-OR provides a direct non-inversion on the remaining input and a "1" input provides an inversion as shown below:



The INSTRUCTIONS are set by Switches G, H, J as shown below:

INSTRUCTION LIST

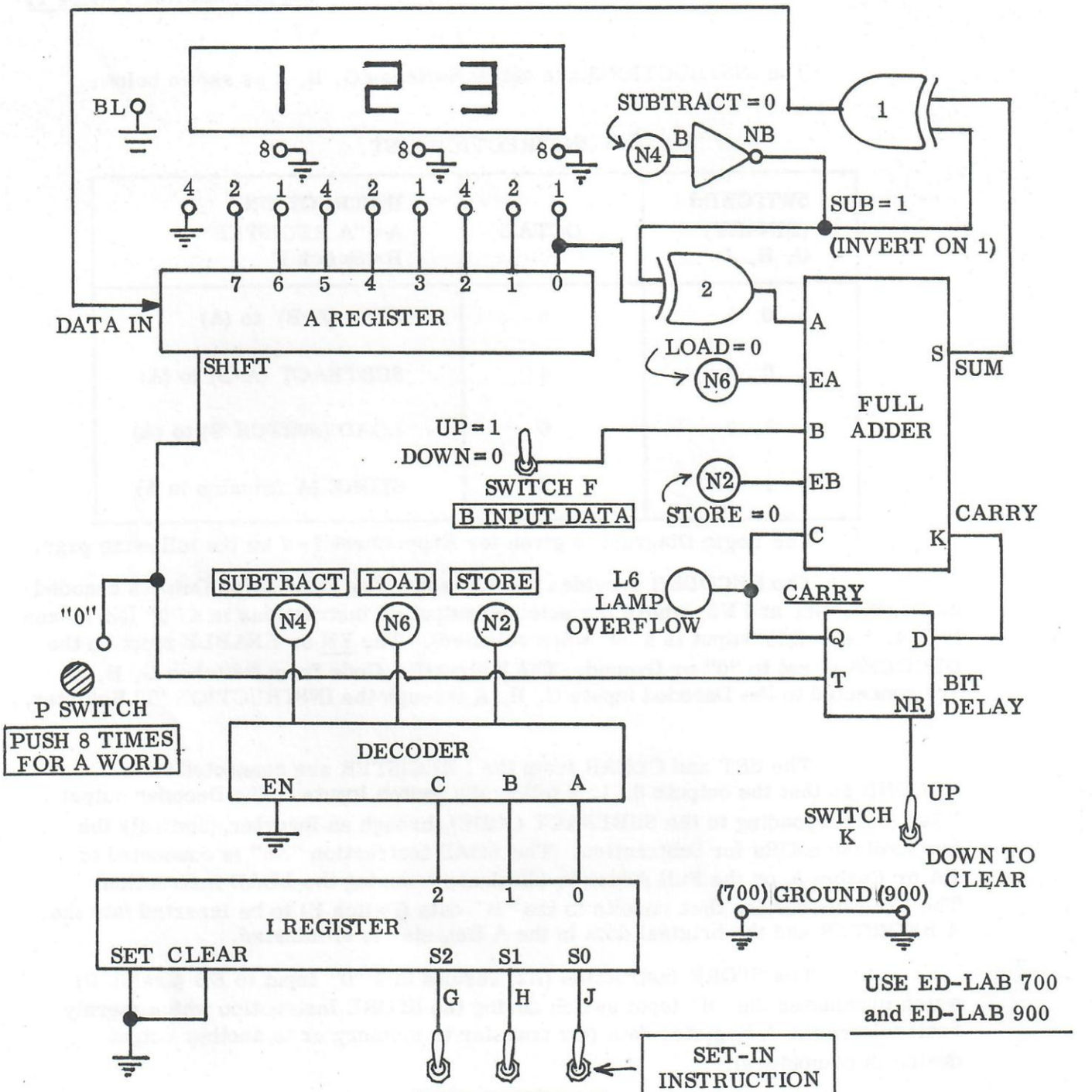
SWITCHES (BINARY) G, H, J	OCTAL	INSTRUCTION A = "A REGISTER" B = Switch F
1 0 1	5	ADD (A+B) to (A)
1 0 0	4	SUBTRACT (A-B) to (A)
1 1 0	6	LOAD (SWITCH B) to (A)
0 1 0	2	STORE (A remains in A)

The Logic Diagram is given for Experiment 7 - 7 on the following page.

The DECODER provides the corresponding Instruction Outputs decoded as N5, N4, N6, and N2, where the selected output or instructions is a "0" (N4 means NOT 4, i.e., this output is a "0" when selected). The EN or ENABLE input on the DECODER is set to "0" or Ground. The Instruction Code from Switches G, H, J are connected to the Decoded inputs C, B, A through the INSTRUCTION "I" Register.

The SET and CLEAR from the I REGISTER are connected to GROUND so that the outputs 0, 1, 2 follow the Switch Inputs. The Decoder output "N4" (corresponding to the SUBTRACT CODE), through an Inverter, controls the two Exclusive-ORs for Subtraction. The LOAD instruction "N6" is connected to EA or Enable A on the Full Adder to eliminate A during the LOAD instruction. The Load Instruction thus results in the "B" data (Switch F) to be inserted into the A REGISTER and the original data in the A Register is eliminated.

The STORE instruction (N2) results in a "0" input to EB (Enable B) which eliminates the "B" input switch during the STORE Instruction which merely recirculates the A Register data (for transfer to memory or to another output device in computers).



SWITCHES →	G	H	J		
	1	0	1	= 5 ADD	(A+B) INTO (A)
	1	0	0	= 4 SUBTRACT	(A-B) INTO (A)
	1	1	0	= 6 LOAD	(B INTO A)
	0	1	0	= 2 STORE	(A REMAINS IN A)

NOTE: FOR UNITS WITH 12 KEY KEYBOARD
The BL pin must be removed from GROUND to turn the DISPLAY ON.

EXPERIMENT 7 - 7 INSTRUCTION ARITHMETIC UNIT

PROCEDURE (Instruction Arithmetic Unit):

1. Wire the Logic in Experiment 7 - 7.
2. Set Switch K DOWN then UP to CLEAR the BIT DELAY "CARRY".
3. Set the following INSTRUCTIONS into the controlled Arithmetic Unit using Switches G, H, J, as given in the table below. After each instruction is set on the Switches, enter the B data using SWITCH F. Set Switch F (1 or 0) and then press Switch P. Repeat this eight times for each binary bit of the 8 bit octal words. The table shows the Instruction or Command in words and Binary Code, the Data in OCTAL and Binary, and the expected Register output after each word step (each 8 pulses).

COMMAND			DATA "B" (SWITCH F)		A REGISTER	CHECK
	Octal	Switch GHJ Binary	Binary	Octal	DISPLAY	
LOAD	6	1 1 0	00000100	004	004	
LOAD	6	1 1 0	00000111	007	007	
ADD	5	1 0 1	00100011	043	052	
ADD	5	1 0 1	01010011	123	175	
SUBTRACT	4	1 0 0	00100100	044	131	
STORE	2	0 1 0	11010100	324 *	131	

* (Note that the STORE Instruction ignores the "B" data)

4. Do the Register outputs agree with the expected results?_____
5. Repeat the Sequence of Instructions or PROGRAM given below. Be sure to set the Command in binary (Switches GHJ) and to insert the B data in binary (Switch F).

PROGRAM: (W+X-Y-Z)=ANSWER

SET-IN ↘ ENTER ↘

COMMAND: Set Switches G, H, J	DATA "B": Set Switch F(Serially)	DISPLAY A Register	CHECK
LOAD (W) 110	074	074	
ADD (X) 101	116	212	
SUB (Y) 100	024	166	
SUB (Z) 100	031	135 (ANSWER)	

6. Do the Register outputs agree with the expected results?_____

continue next page.....

DESCRIPTION (Memory and Arithmetic Unit):

The Logic given in this Experiment 7 - 8 shows a combination of the Arithmetic Logic from Experiment 7 - 7 together with the Memory and Address Counter Logic from Experiment 4 - 3. The result is shown in the diagram for Experiment 7 - 8 on the following page which is the Arithmetic Unit with Instruction Control together with the Memory which stores both the DATA and INSTRUCTIONS.

With reference to the diagram on the following page, the Counter is connected to the Memory Address. Switch F, which provides the input data, is connected to the Memory DATA IN. The Memory data output (OUT A) is connected to both the "I REGISTER" and "A REGISTER" DATA IN. Data is transferred to a Register by activating or gating the SHIFT pulses. Gate AND #3 is ON for "Even" Memory addresses for Shifting to the I REGISTER. We have arbitrarily selected EVEN ADDRESS (i.e., 0, 2, 4 etc.) for INSTRUCTIONS. Gate AND #1 is ON for ODD Memory Addresses which then correspond to DATA words. ODD and EVEN Addresses are determined by using the 3-bit on the Address Counter. Shifting (T) into the BIT DELAY for Adder Carry uses the same pulse as the "A Register" SHIFT.

The complete operation consists of first transferring a word to the I REGISTER with the INSTRUCTION in an INSTRUCTION CYCLE. This occurs by the operator depressing the P SWITCH 8 times. The "3-bit" from the Address Counter then becomes a "1" and the next word (ODD) is transferred to the "FULL ADDER Input B" and then to the A REGISTER. This is the OPERATION CYCLE.

Switches A and C and D on the Ed-Lab 700 are used in initially placing the data into Memory.

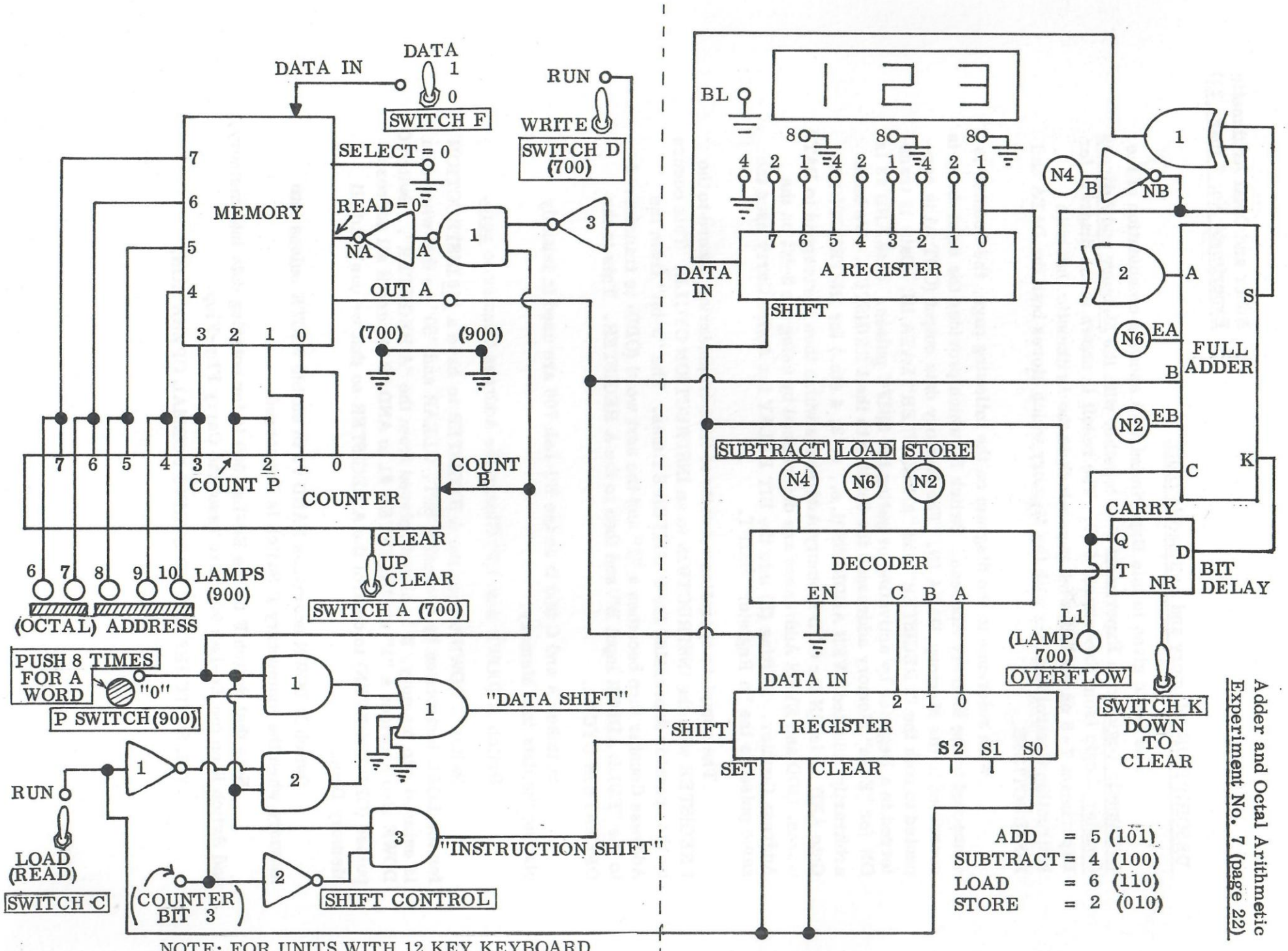
Switch A (DOWN then UP) Clears the Address Counter to zero.

Switch C DOWN forces the A REGISTER to have a 110 INSTRUCTION for the LOAD instruction by grounding SET, CLEAR and "S0" so that when data is entered into memory, it is also displayed from the "A REGISTER". Switch C DOWN also provides a "1" via INTERTER #1 to AND #2 to permit all address pulses (ODD and EVEN) to control the A REGISTER so that we can read all Memory Data.

Switch D DOWN energizes NAND #1 to enable WRITE pulses into memory when the momentary P Switch is Depressed.

Note that Switch F on the Ed-Lab 900 is for entering data into memory, and Switch K on the Ed-Lab 900 is to reset the Carry Flip-Flop.

ALL SWITCHES ARE UP DURING NORMAL OPERATION.



NOTE: FOR UNITS WITH 12 KEY KEYBOARD
The BL pin must be removed from GROUND to turn the DISPLAY ON.

PROCEDURE:

1. Wire Experiment Logic 7-8.

   If Experiment Logic 7-7 has been wired, you can remove 6 wires, and then continue to complete this logic.

   (a) Remove 2 wires from SWITCH P ("0")
   to "A REGISTER" SHIFT
   to "BIT DELAY" T (TRIGGER)

   (b) Remove 3 wires from SET and CLEAR
   to GROUND on "I REGISTER"

   (c) Remove 1 wire from "SWITCH F"
   to "FULL ADDER" B INPUT

   Then add MEMORY and ADDRESS COUNTER and SHIFT CONTROL wires as in Experiment Logic 7-8.

2. PROGRAM

   The program to be run is similar to that run previously:

   ANSWER = V + W + X + Y − Z

   The program instructions and data are given below:

Instructions	Data	A REGISTER	
LOAD (V)	007	007	
ADD (W)	014	023	
ADD (X)	042	065	
ADD (Y)	201	266	
SUB (Z)	135	131	
STORE	234	131	ANSWER

   The Memory alternately stores the Instructions and data as given on the following page:

ENTER MEMORY ↓

Memory	Octal Data	A Register
LOAD	006	
007	007	007
ADD	005	remains 007
014	014	023
ADD	005	023
042	042	065
ADD	005	065
201	201	266
SUB	004	266
135	135	(answer) 131
STORE	002	131
234	234	131

3. DATA ENTRY

The first step is to enter the data into memory.

Set Switch D DOWN (for WRITE).

Set Switch C DOWN (for LOAD or "READ").

Set Switch A DOWN then UP (to reset Address).

Set Switch K DOWN then UP (to reset Carry).

(The Carry is monitored on LAMP 1/Ed-Lab 700 and should always be zero during data entry.)

Enter Memory Data using Switch F (1 and 0) and then depressing Switch P; one bit at a time for all words as given in the ENTER column in the above table.

4. READ DATA

Set Switch D UP (there is no more writing).

Keep Switch C DOWN (for reading).

Set Switch A DOWN then UP (to reset address).

Keep depressing Switch P. Every 8 pulses should result in the next data word being displayed. Are all data words properly displayed?_____

5. RUN PROGRAM

Set Switch A DOWN then UP (to reset address).

Set Switch C UP (for normal run).

(Switch D should remain UP).

ALL SWITCHES WILL BE UP.

Depress the P Switch 8 times for the INSTRUCTION CYCLE. The Display will not change.

Depress the P Switch 8 times for the OPERATION CYCLE.

The Display should have 007.

Keep depressing the P Switch as above; 8 for the I REGISTER and 8 for the A REGISTER.

Does the A REGISTER display the correct results?_____

6. SUMMARY

The above Experiment 7 - 8 has combined the ADDER, REGISTERS, MEMORY and CONTROL for a Programmed Computer where the operator generates the Timing using the P Switch. Four instructions were used for LOAD, ADD, SUBTRACT, and STORE. This computer was not designed for writing into Memory with the program.

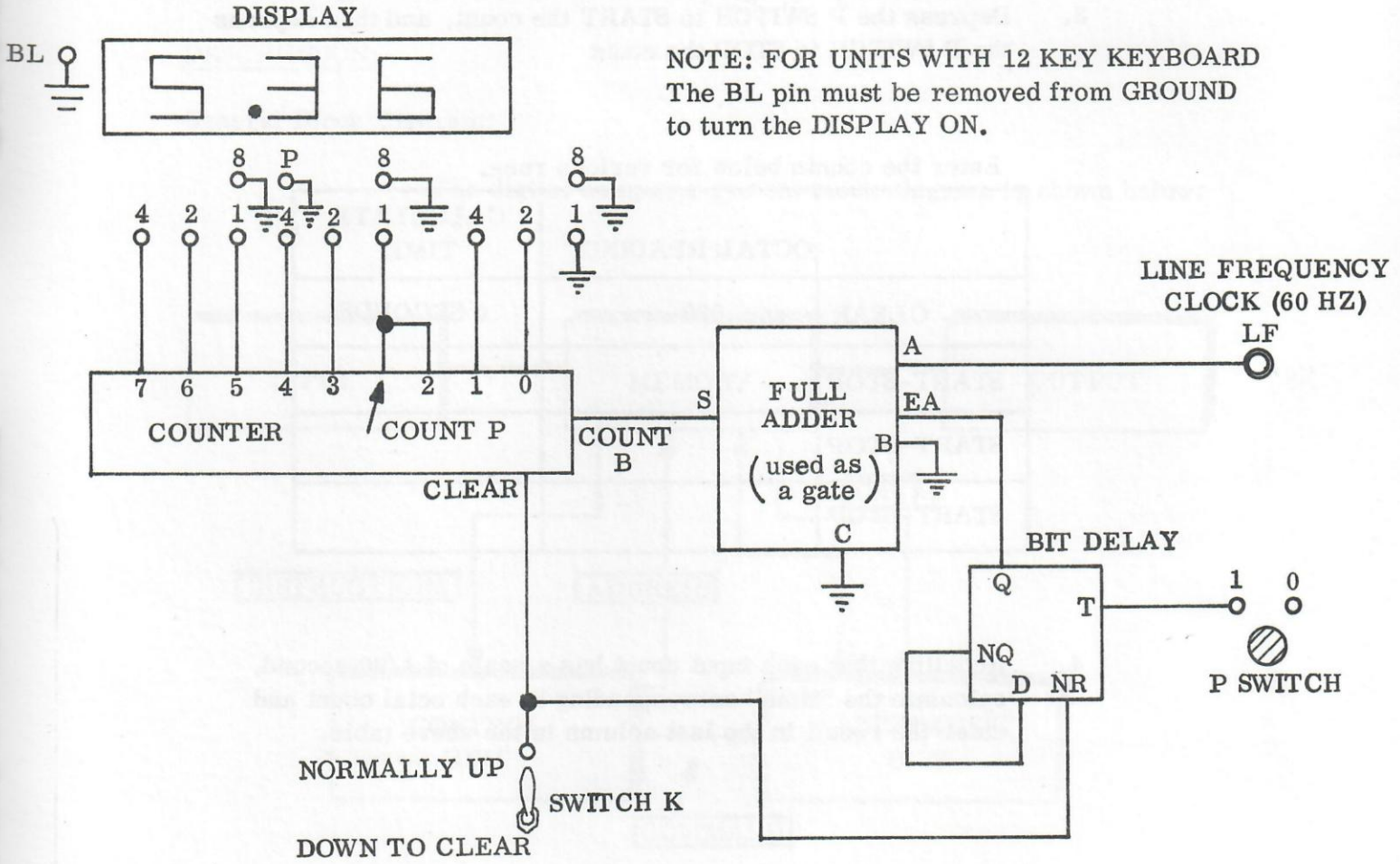
In the following Experiment 8, a complete Automatic Stored-Program Computer is implemented. Eight instructions are used. The additional four instructions include: TRANSFER (Change Address of next Instruction), TRANSFER on NEGATIVE (Change Address of next Instruction only on Negative Number in A Register; i.e, "1" in most significant bit), SHIFT RIGHT, and Automatic HALT.

The STORE Instruction will transfer data to Memory. The 5 bits in the Instruction word will specify the Address of the Data so that the Instructions and Data can be in any location (not ODD or EVEN only). The automatic timing pulses will also be generated in Experiment 8.

continued next page.....

DESCRIPTION (Line Frequency Counter) :

The latest models of the Ed-Lab 900 have a Line Frequency Clock signal available at the terminal jack labeled LF on the upper righthand portion of the panel. The Logic Diagram is shown below for Experiment 7 - 9.



Experiment 7 - 9 LINE FREQUENCY COUNTER

This experiment uses the LF Clock connected to the counter through the Full Adder which is used merely as an AND gate with A and EA (enable A) as inputs, and the Sum S as an output. The BIT DELAY is used as a standard Triggered Flip-Flop by connecting the "NG" output to the "D" input. Switch K (DOWN then UP) clears the "PROGRAM COUNTER" and the BIT DELAY Flip-Flop. Depressing the P SWITCH will alternately "Toggle" the Flip-Flop and thus START and STOP the Counting. Counting is in Octal and each input Count B is with a time interval of 1/60 SECOND.

PROCEDURE (LINE FREQUENCY COUNTER) :

1. Wire the logic as in Experiment 7 - 9.
2. Set Switch K DOWN then UP to Clear the Counter and to Clear the Control Flip-Flop (BIT DELAY).
3. Depress the P SWITCH to START the count, and then depress the P SWITCH to STOP the count

Enter the counts below for various runs.

	OCTAL READING	CALCULATE TIME
CLEAR	000	0 SECONDS
START-STOP		
START-STOP		
START-STOP		

4. Recalling that each input count has a scale of 1/60 second, calculate the "time" corresponding to each octal count and enter the result in the last column in the above table.

end of experiment #7.....

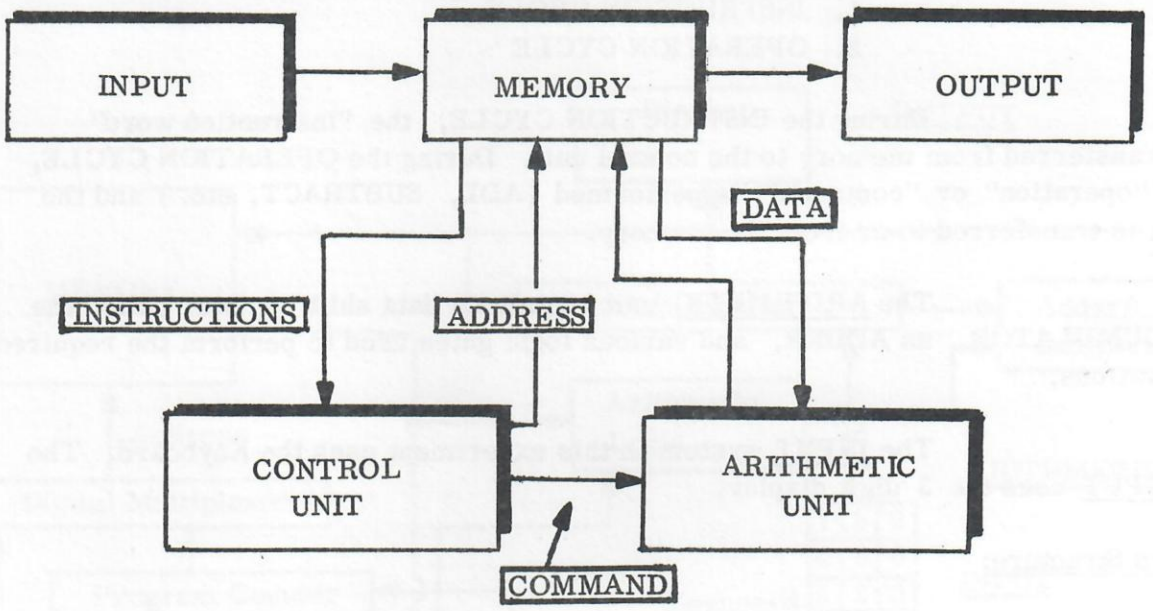
OBJECTIVE:

This experiment uses one COMPUTER MEMORY & DISPLAY LAB, (Ed-Lab 900) and one COMPUTER SYSTEMS LAB (Ed-Lab 700) to form a complete AUTOMATIC DIGITAL COMPUTER with continuous automatic and operator single-step control and stored program operation.

DESCRIPTION:

General Block Diagram:

The digital computer system block diagram is shown below:



GENERAL BLOCK DIAGRAM
Automatic Digital Computer

The MEMORY stores two types of words;

1. DATA
2. INSTRUCTION

The DATA words are the numbers to be used in the problem.

The INSTRUCTION words have two parts denoted as ADDRESS and COMMAND (or OPERATION). The ADDRESS specifies the location in MEMORY and the COMMAND denotes the "operation" to be performed. Thus an Instruction such as:

INSTRUCTION

24 ADD

specifies that the "number or DATA WORD located in memory address 24" is to be ADDED to the number in the ARITHMETIC UNIT.

The CONTROL unit generates all of the timing signals and controls the two timing cycles of the computer:

1. INSTRUCTION CYCLE
2. OPERATION CYCLE

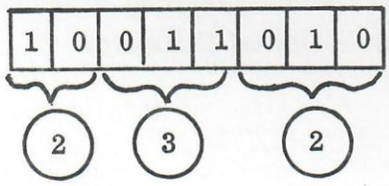
During the INSTRUCTION CYCLE, the "instruction word" is transferred from memory to the control unit. During the OPERATION CYCLE, the "operation" or "command" is performed (ADD, SUBTRACT, etc.) and the data is transferred to or from the memory.

The ARITHMETIC unit contains a data shift register called the ACCUMULATOR, an ADDER, and various logic gates used to perform the required operations.

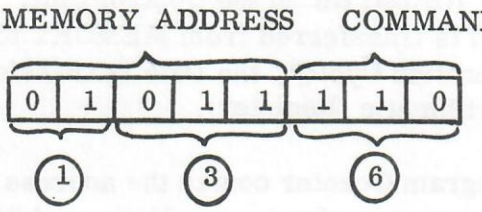
The INPUT system in this experiment uses the Keyboard. The OUTPUT uses the 3 digit display.

Word Structure:

The 256 bit memory is organized in 8 bit words with a total of 32 words. The data words have 8 bits. The binary bits can be represented in "octal" as shown below:



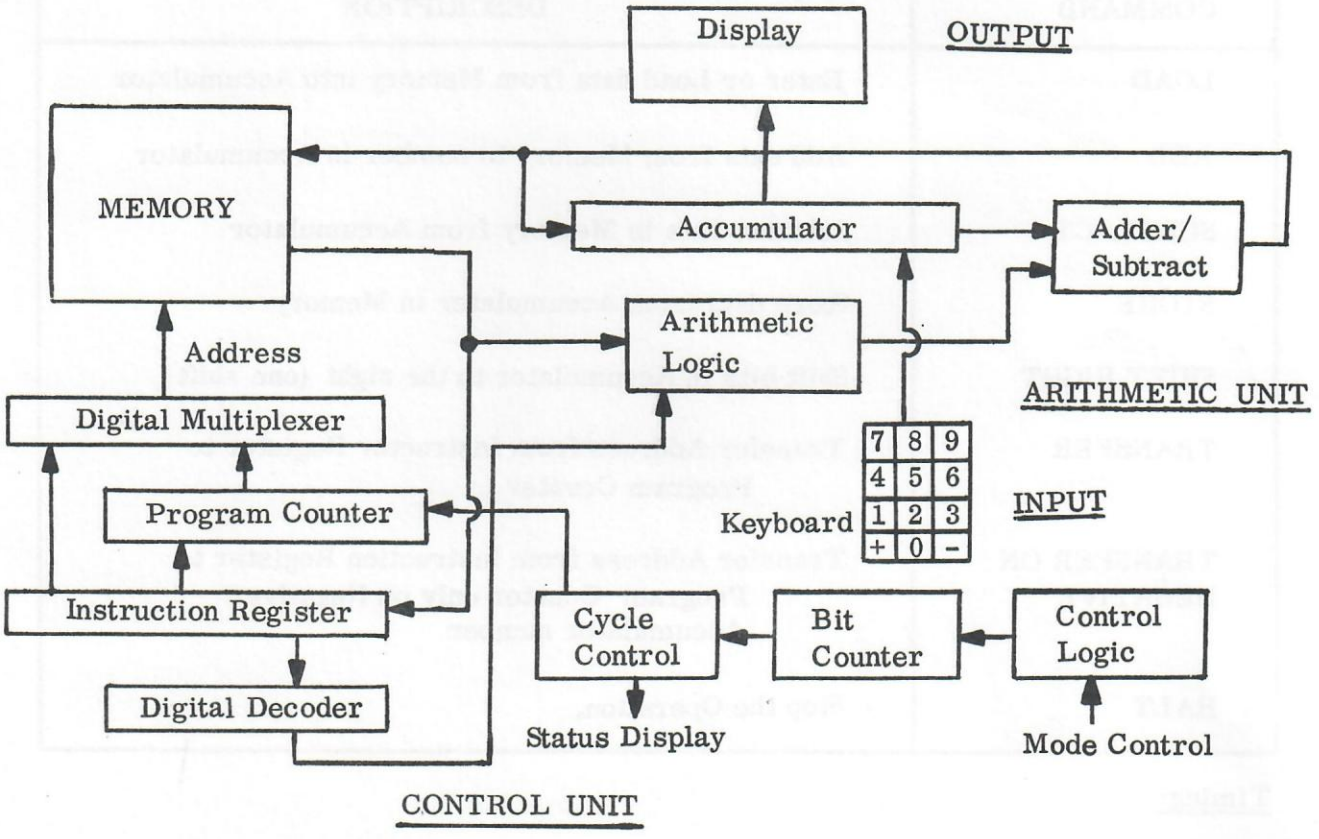
The instruction words with 8 bits is coded as follows:



There are 8 commands built into this computer and they are adequate to perform any computer problem. The five bits of memory address specify the 32 different address locations.

Detailed Block Diagram:

The detailed Block Diagram is shown below:



COMPUTER BLOCK DIAGRAM

The two Shift Registers are the ACCUMULATOR in the Arithmetic Unit and the INSTRUCTION REGISTER in the Control Unit. During the "Instruction Cycle" the Instruction word is transferred from MEMORY to the Instruction Register. During the "Operation Cycle", the Data is transferred to or from memory to the Arithmetic Unit (Arithmetic Register).

The Program Counter counts the address location of the Instruction Word. The Digital Multiplexer transfers, as a Memory Address, either the number from the Instruction Register (during the Operation Cycle) or the number from the Program Counter (during the Instruction Cycle). The Digital Decoder decodes the 3 bits of command data into eight signal lines for each of the eight "commands" used.

Commands (Operations):

The eight commands are listed below:

COMMAND	DESCRIPTION
LOAD	Enter or Load data from Memory into Accumulator
ADD	Add data from Memory to number in Accumulator
SUBTRACT	Subtract data in Memory from Accumulator
STORE	Store data from Accumulator in Memory
SHIFT RIGHT	Shift bits in Accumulator to the right (one shift)
TRANSFER	Transfer Address from Instructor Register to Program Counter
TRANSFER ON NEGATIVE	Transfer Address from Instruction Register to Program Counter only on Negative Accumulator number
HALT	Stop the Operation.

Timing:

The basic timing uses a "four-phase" clock; followed by "eight pulses" corresponding to the 8 word bits (via a 3 Bit Counter); followed by an alternating (Instruction/Operation) Cycle Control; followed by an ON-OFF Control.

Programming:

The computer operation consists of having a sequence of instuctions direct operations to solve the required problem. Consider the problem of adding two numbers:

$$X = A + B$$

Assume that A is stored in memory location 17; B is stored in 25; and X is desired to be stored in 31 when calculated. Assume that the first instruction (Program Counter) starts at location 00. The required sequence of instructions is then:

PROGRAM COUNTER	INSTRUCTION		COMMENTS
	ADDRESS	COMMAND	Contents of Accumulator
00	17	LOAD	A
01	25	ADD	A + B = X
02	31	STORE	A + B = X
03	-	HALT	

The sequence of instructions is called the PROGRAM. The computer operator is then instructed as follows:

Operator Instructions

TO ADD X = A + B
1. Enter A in 17
2. Enter B in 25
3. Start Program at 00
4. Answer X results in 31

Computer Logic:

The Automatic Digital Computer is constructed in three stages shown in Figure 8-1A, 8-1B and 8-1C.

Figure 8-1A is the Timing Logic. The clock output drives a "phase clock" consisting of Flip-Flop #1 and #2 connected in a shift register configuration. This generates the timing signals for the computer operations. Signal B1 drives a "BIT COUNTER" (Flip-Flops #3, #4 and #5), which generates the 8 binary counts needed to address the 8 bits of each word in the Memory. A signal called BIT "0" is generated at the end of each 8 counts by NOR #1. Flip-Flop #8 provides alternating INSTRUCT and OPERATE signals to activate the INSTRUCT and OPERATE functions of the computer. Momentary switch "M" provides the START signal when pressed, via Flip-Flop #6, #7 and #9. Flip-Flop #9 generates the ON and OFF signals.

Figure 8-2 and 8-3 show the timing signals generated by this logic.

Figure 8-1B is the Instruction Logic. During each Instruct Cycle the Instruction word from memory is loaded into the Instruction Register. The bits 0, 1 and 2 of the Instruction Word in the instruction register drive the digital decoder inputs and select the Decoder Output which corresponds to the present Instruction. Note that the Decoder Output will be selected only when the Decoder Enable input (EN) is "Low". This occurs during the Operate Cycle. During the Operate Cycle the logic gates in the Instruction Logic Diagram route the correct signals to the computer Counter, Registers, Adder and Bit Delay to execute the instruction. The timing diagrams for each signal generated by each Instruction will be measured in this experiment.

When Switch F is Down, the Instruction is set manually with switches GHJ. This allows the operator to program the computer by storing and reading out instructions from memory. A summary of each instruction and the logic signals it generates is given below:

TRANSFER ON NEGATIVE:

Instruction Code 0 (Transfer on Negative) will cause Decoder Output N0 to go "low". If the A. Register N7 output is "low" also, indicating a negative number is in the A. Register, then NOR #2 generates a Transfer on Negative signal via OR #3 into NAND #1. The timing signals A1, B2 and BIT "0" and the transfer signal from OR #3 generate a negative going pulse during the OPERATE cycle which SETS the Program Counter to the Address specified in the Instruction. Thus the Transfer on Negative instruction causes the Address in the Instruction to be transferred to the Memory via the Program Counter when a negative number is in the A. Register.

TRANSFER:

Instruction code 1 (Transfer) will cause Decoder Output N1 to go "low". This generates the same "Transfer" operation as Transfer On Negative via the Inverter #3 driving OR #3. The difference is that Transfer will cause the Program Counter to change to the Address in the Instruction regardless of the value of the number in the A. Register.

STORE:

Instruction code 2 (Store) will cause Decoder Output N2 to go "low". This store signal drives NAND #3 via Inverter #2. NAND #3 is also driven by timing signals A1 and B2 to generate a WRITE signal into the READ/WRITE input to the Memory via the Inverter D on the Ed-Lab 900. The STORE Instruction places the data in the A. Register into the computer memory at the memory address specified by the instruction.

SHIFT RIGHT:

Instruction code 3 (Shift Right) will cause Decoder Output N3 to go "low". The "low" from N3 allows the BIT "0" pulse to drive AND #4 via OR #1. AND #4 combines the BIT "0" pulse generated by OR #1 during a Shift Right Instruction with the OPERATE CYCLE signal and the Timing Pulses A1 and B1 via AND #2. The output of AND #4 generates a single Shift Pulse into the A. Register Shift input via OR #2. OR #2 allows the signal from the "P" switch to generate shift pulses. Thus the "P" switch may be used to manually generate Shift Pulses. The Shift Right instruction causes the data in the A. Register to be shifted one place to the right.

SUBTRACT:

Instruction code 4 (Subtract) will cause Decoder Output N4 to go "low". N4 is inverted by NOT #1 and the resulting signal is used to implement the Subtract function. The details of the subtract instruction logic are covered in experiment number 9. The Subtract instruction causes the number in the memory location contained in the instruction to be subtracted from the number in the A. Register and the result is stored in the A. Register.

ADD:

Instruction code 5 (Add) will cause Decoder Output N5 to go "low". This signal is not connected to any other logic. In the absence of any other instruction the computer will perform the Add operation. The Add instruction causes the number in the memory location specified in the Instruction to be Added to the number in the A. Register and the resulting SUM to be stored in the A. Register.

LOAD:

Instruction code 6 (Load) will cause Decoder Output N6 to go "low". This places a "low" on the Full Adder EA enable input, disabling the A input to the Full Adder. During the execute cycle this causes the Add operation to be performed with the exception that the data from the A Register is not accepted by the Full Adder. The machine will add the data in the memory location contained in the instruction to zero and load the result in the A Register. The Load instruction causes the data in the memory location indicated by the instruction to be Loaded into the A Register.

HALT:

Instruction code 7 (Halt) will cause the Decoder Output N7 to go "low". This "low" will generate a "high" at the output of NAND #2, if Switch E is in the UP (Auto) position. The "high" on the Trigger of Flip-Flop #9 permits the OP signal to turn the Flip-Flop OFF via D3 (see Figure 8-1A).

COMPUTER LABELS:

It is convenient to note the coding of the switches and lamps on the Ed-Lab 700 and Ed-Lab 900. You can place a piece of paper or card beneath the switches and lamps with the labels as shown in Figure 8-4.

WIRING PROCEDURE OUTLINE:

The complete computer is wired in 3 STEPS.
The first step is to wire and test the TIMING LOGIC Figure 8-1A.
The second step is to wire the logic for the INSTRUCTION LOGIC (Figure 8-1B). It is not necessary to test this logic before completing the wiring for the rest of the system. The Instruction Logic can be tested as part of the complete computer. However, in order to test anything, it is necessary to have correct timing signals.

The third step is to wire the MEMORY and ARITHMETIC Logic (Figure 8-1C). This third step is performed in Experiment No. 9. In this Experiment No. 8, steps 1 and 2 will be wired. The TIMING will be tested without an oscilloscope. As an option and only if desired, the timing can be tested with an Oscilloscope and the Instruction Logic can be tested with an Oscilloscope. If the logic does not operate properly, then troubleshooting can be studied. The instructor can also introduce troubles for the student to detect and repair.

PROCEDURE:

1. (Step 1) Wire the TIMING LOGIC given in Figure 8-1A. Use only the Ed-Lab 700.

2. Set Switch E DOWN to the ONE CYCLE position.

3. Lamps 1 and 3 should both be LIT indicating INST Cycle (is ready and the system is OFF.

4. Test the ONE CYCLE operation. Set the CLOCK SWITCH TO HIGH (full RIGHT) 1 KHz Speed.

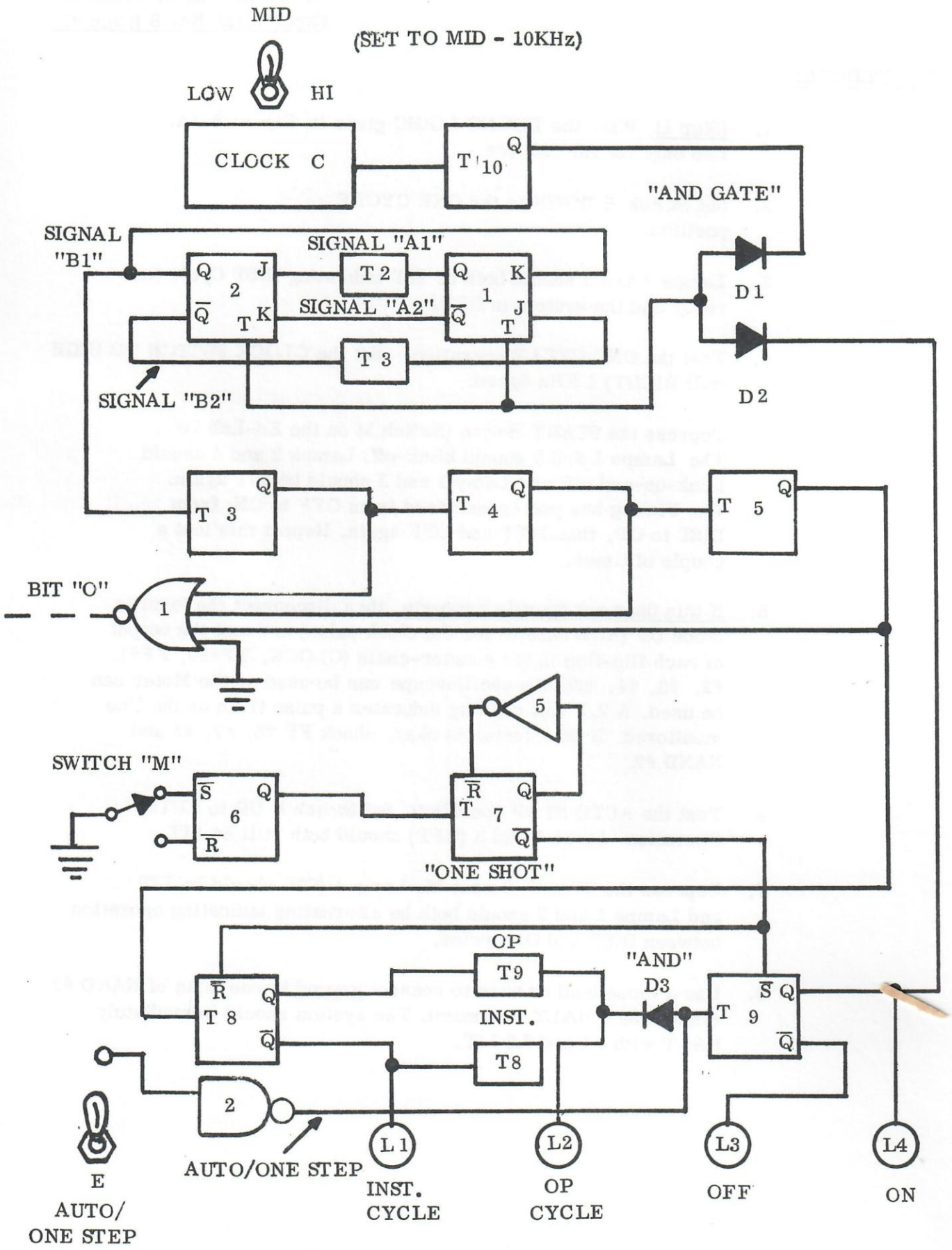
   Depress the START Button (Switch M on the Ed-Lab 700). The Lamps 1 and 3 should blink-off; Lamps 2 and 4 should blink-on-and off; and Lamp 1 and 3 should be LIT again. The Timing has just transferred from OFF to ON; from INST to OP, then INST and OFF again. Repeat this test a couple of times.

5. If this does not operate properly, then disconnect the input to diode D2 (AND control for the clock pulse) and test the output of each flip-flop in the counter-chain (CLOCK, FF#10, FF#1, #2, #3, #4, #5). An oscilloscope can be used or the Meter can be used. A 2.5 volt reading inducates a pulse train on the line monitored. If the counter is okay, check FF #6, #7, #8 and NAND #2.

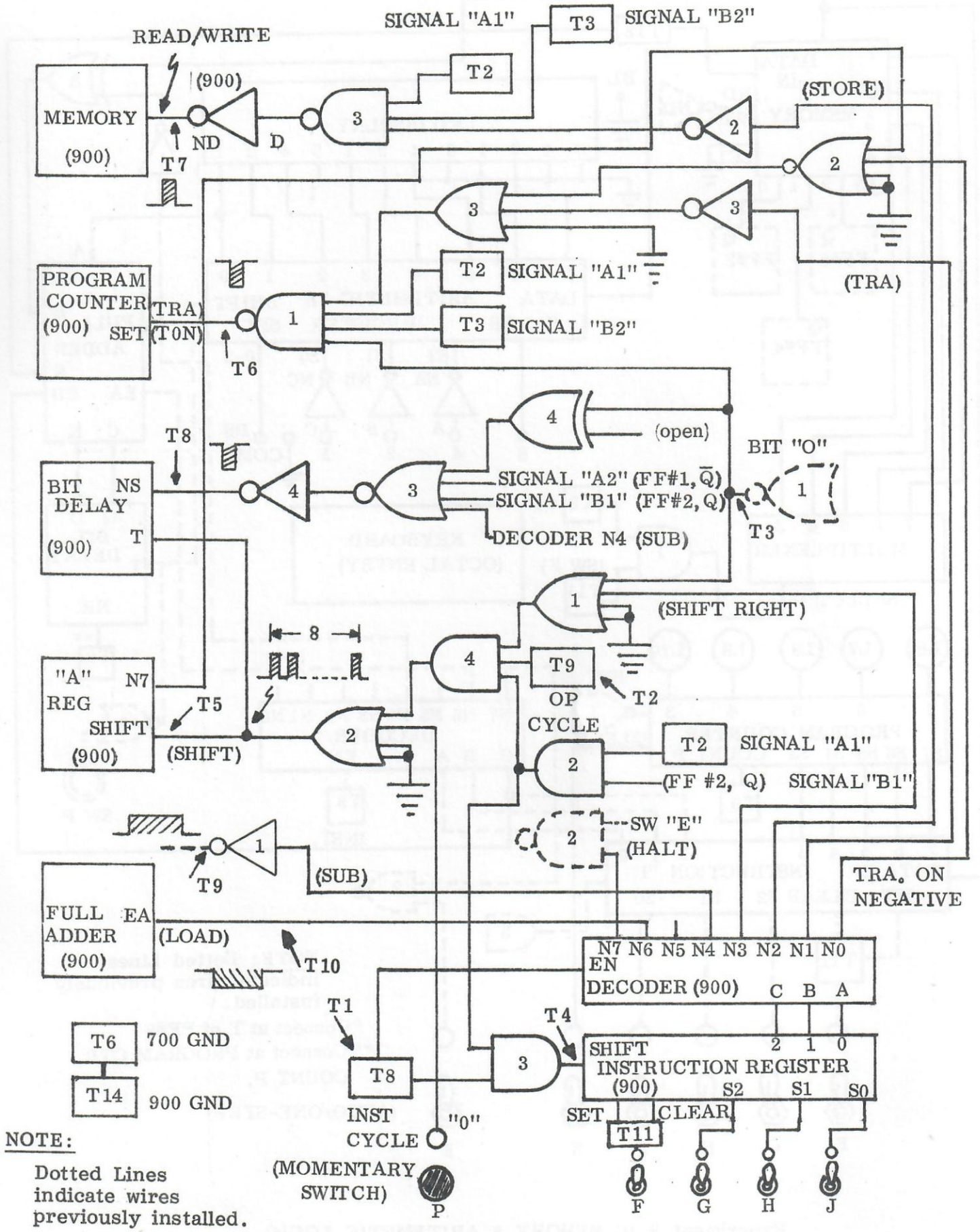
6. Test the AUTO STOP operation. Set Switch E UP to AUTO Operation. Lamp 1 and 3 (OFF) should both still be LIT.

7. Depress the START Button M. Lamp 4 (ON) should be LIT and Lamps 1 and 2 should both be alternating indicating operation between INST and OP cycles.

8. Use a loose lead or wire to connect ground to one input of NAND #2 simulating a HALT Command. The system should immediately HALT with L1 and L3 LIT.



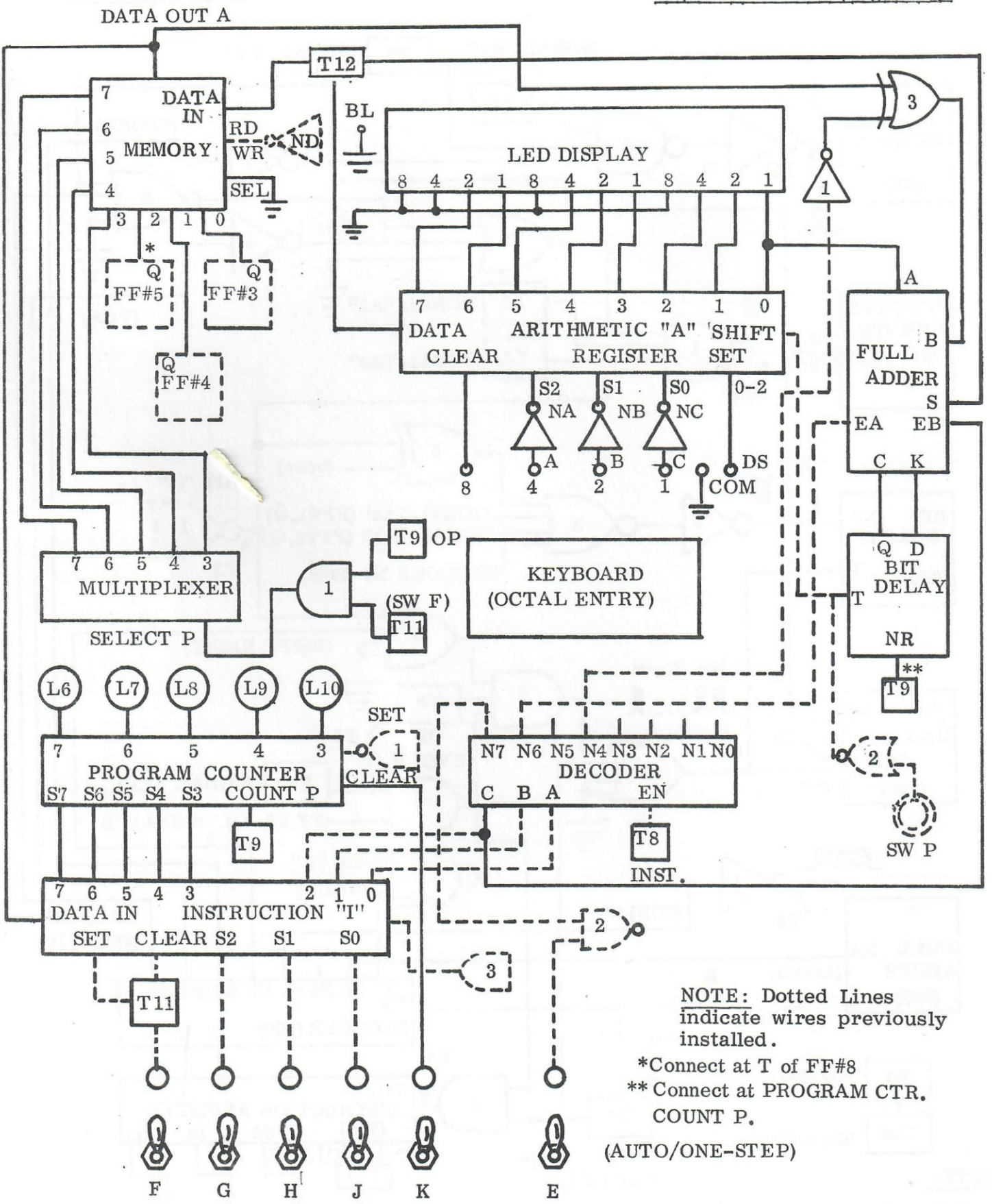
Experiment 8-1A TIMING LOGIC



NOTE:

Dotted Lines indicate wires previously installed.

Experiment 8-1B INSTRUCTION LOGIC DIAGRAM



NOTE: Dotted Lines indicate wires previously installed.

*Connect at T of FF#8

** Connect at PROGRAM CTR. COUNT P.

(AUTO/ONE-STEP)

Experiment 8-1C MEMORY & ARITHMETIC LOGIC

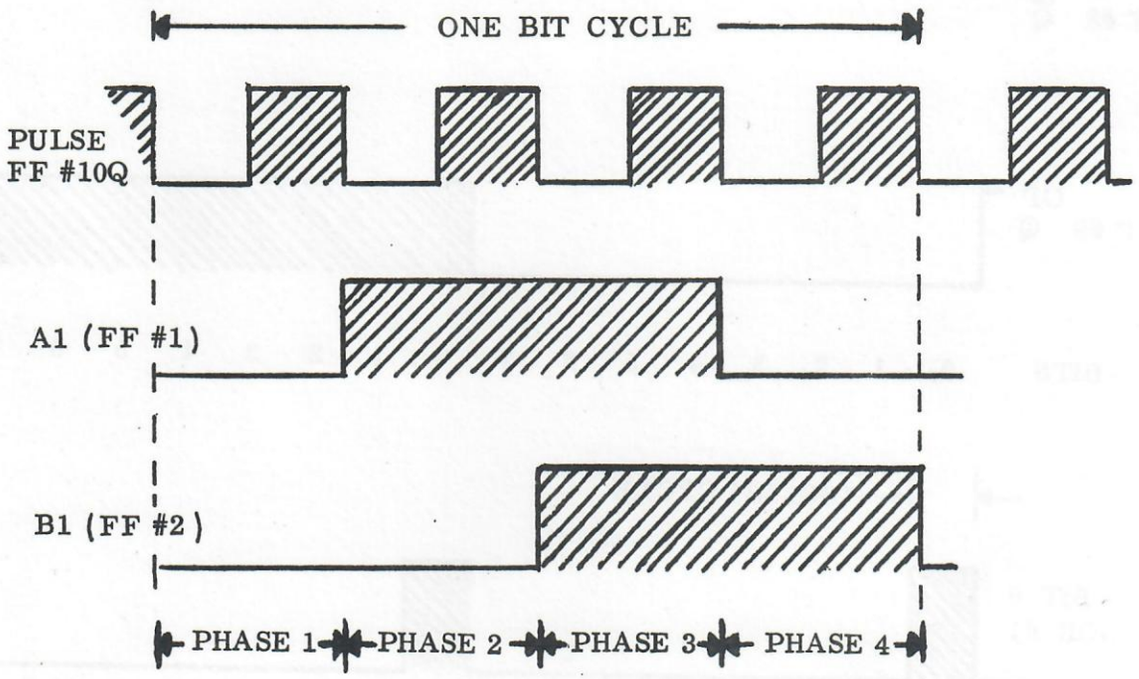


Figure 8-2 BIT TIMING WAVEFORM

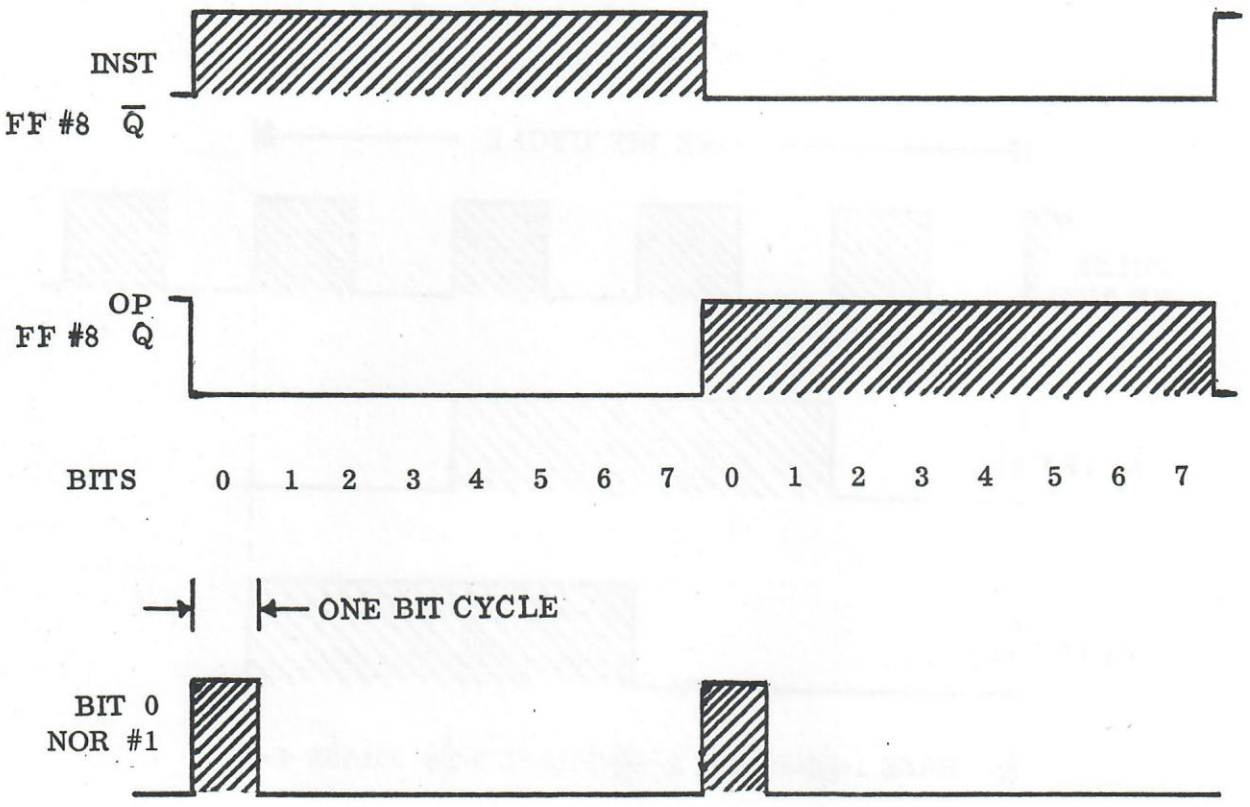
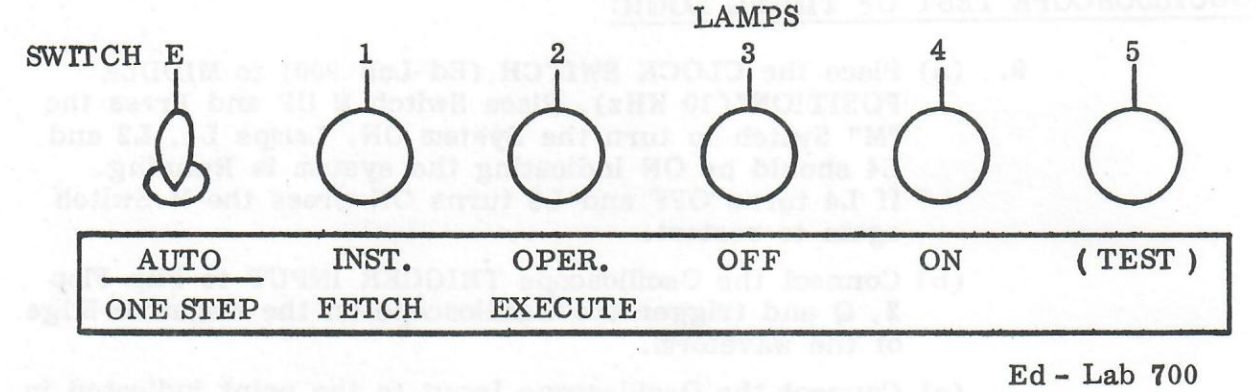
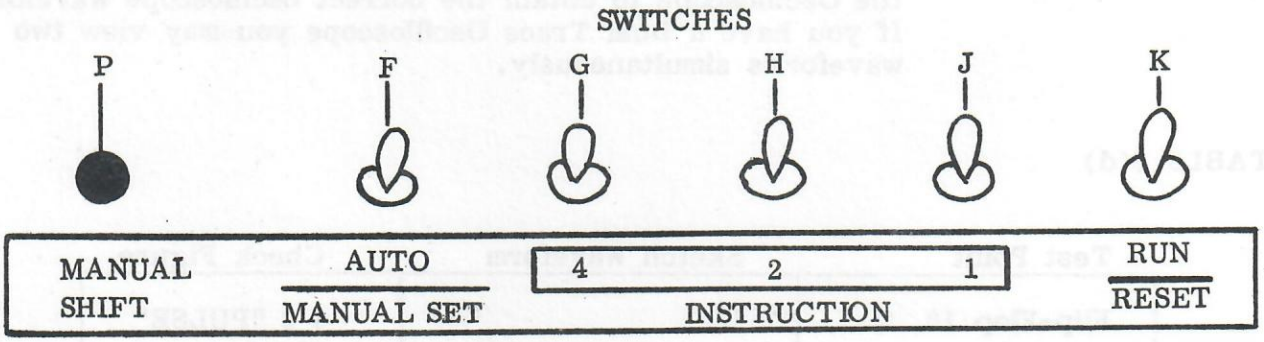


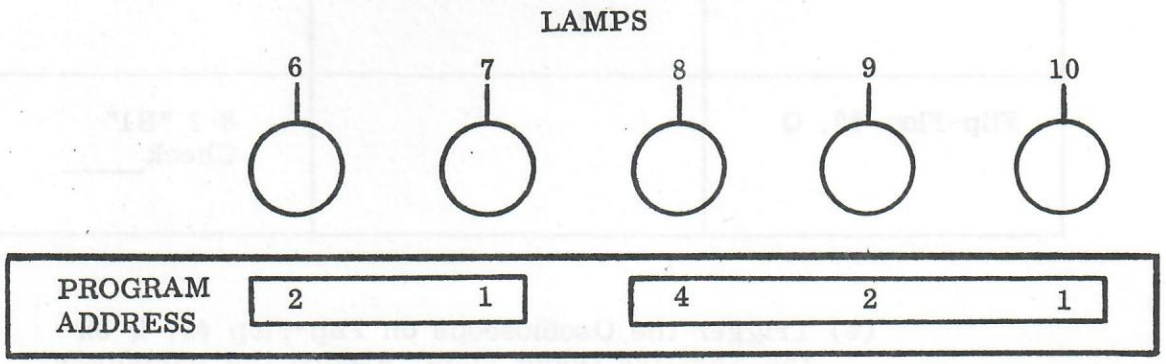
Figure 8-3 WORD and CYCLE TIMING WAVE



Ed - Lab 700



Ed - Lab 900



Ed - Lab 900

Figure 8-4 COMPUTER LABELS (EXACT SIZE)

OSCILLOSCOPE TEST OF TIMING LOGIC

9. (a) Place the CLOCK SWITCH (Ed-Lab 900) to MIDDLE POSITION*(10 KHz). Place Switch E UP and Press the "M" Switch to turn the System ON. Lamps L1, L2 and L4 should be ON indicating the system is Running. If L4 turns OFF and L3 turns ON press the M Switch again to restart.

   (b) Connect the Oscilloscope TRIGGER INPUT to Flip-Flop 2, Q and trigger the Oscilloscope on the Negative Edge of the waveform.

   (c) Connect the Oscilloscope Input to the point indicated in the left hand column of table 9(d) and sketch the Waveform. Compare the observed waveform with the waveform given in the figure indicated. (Note: Adjust the sweep time on the Oscilloscope to obtain the correct oscilloscope waveforms). If you have a Dual Trace Oscilloscope you may view two waveforms simultaneously.

TABLE 9(d)

Test Point	Sketch Waveform	Check Figure
Flip-Flop 10, Q		8-2 "PULSE" Check____
Flip-Flop #1, Q		8-2 "A1" Check____
Flip-Flop #2, Q		8-2 "B1" Check____

   (e) Trigger the Oscilloscope on Flip-Flop #8, Q on the Negative Going Edge. Record the waveforms indicated in the table in 9(f) and check against the waveforms shown in Figure 8-3.

* Note: Earlier trainers may not include the MIDDLE position on the clock switch. If your trainer does not have a MIDDLE position on the clock switch use the HIGH position.

TABLE 9(f)

TEST POINT	SKETCH WAVEFORM	CHECK FIGURE
Flip-Flop #8, $\overline{Q}$		8-3 "INST" Check____
Flip-Flop #8, Q		8-3 "OP" Check____
NOR #1		8-3 "BIT 0" Check____

10. (Step 2) Wire the INSTRUCTION logic circuits in Figure 8-1B. Use the Ed-Lab 700 and the Ed-Lab 900. Be sure to connect ground between the Ed-Lab 700 and Ed-Lab 900.

11. IT IS NOT NECESSARY TO TEST THIS INSTRUCTION LOGIC BEFORE PROCEEDING TO THE NEXT EXPERIMENT.

12.(a) If it is desired to test this portion of the logic, proceed with the steps given below.

TEST POINT (INSTRUCTION)	WAVEFORM
① FF #8, $\overline{Q}$ INST. CYCLE	
② FF #8, Q OP. CYCLE	
③ NOR #1 BIT 0	
④ I REG. SHIFT	
⑤ A REG. SHIFT	
⑥ A REG SHIFT (SHIFT RIGHT)	(SHIFT RIGHT)
⑦ PROGRAM CTR SET (NO TRANSFER)	"1"
⑧ PROGRAM CTR SET (TRANSFER)	TRANSFER
⑨ PROGRAM CTR SET (NO TRANSFER ON NEG)	"1" TRANS ON NEG (NO NEGATIVE)
⑩ PROGRAM CTR SET (TRANSFER ON NEGATIVE)	TRANSFER ON NEGATIVE
⑪ MEMORY READ/WRITE (LOAD)	"0"
⑫ MEMORY READ/WRITE (STORE)	
⑬ BIT DELAY NS (NO SUBTRACT)	"1"
⑭ BIT DELAY NS (SUBTRACT)	
⑮ SUBTRACT (NOT #1) (NO SUBTRACT)	"0"

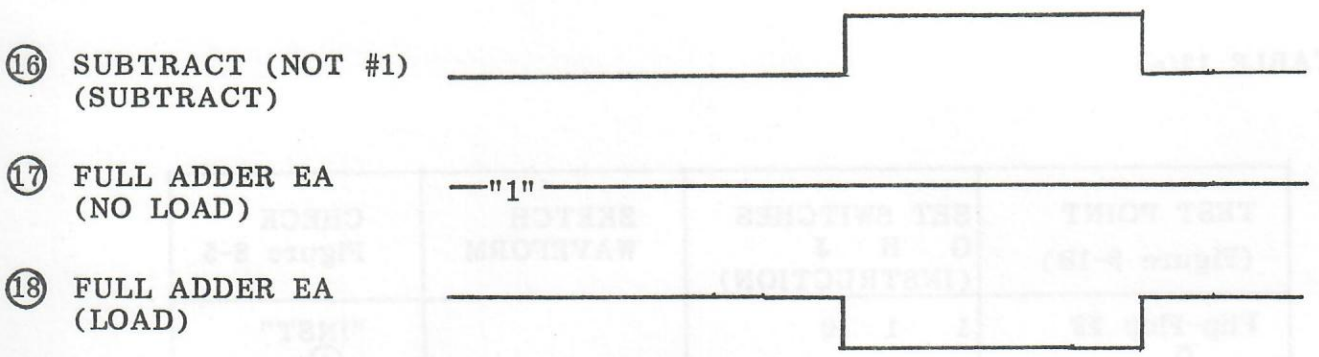


Figure 8-5 INSTRUCTION LOGIC WAVEFORMS

(b) This test consists of running the computer in Auto Mode and Manually operating the instructions to produce the Instruction Signals used in the computer.

i Set CLOCK to MID (10 KHz)*
ii Set SWITCH F (Ed-Lab 900) DOWN for manual Instruction Input
iii Set Switch E UP for AUTO Operation
iv Press M Switch to START the computer.
Lamp L3 will be OFF.
Lamp L4 will be ON.

If the computer STOPS (L3 ON) during this test, restart the Computer by pressing the M Switch.

(c) Trigger the Oscilloscope from Flip-Flop #8, Q and trigger on the Negative Going Edge of the waveform.

(d) Connect the Oscilloscope Vertical Input to the TEST POINTS indicated in Table 12(e). Set Manual Instruction Switches G, H, J to the positions (1 and 0) indicated in table 12(e). Sketch the output waveform and compare to the waveform given in Figure 8-5.

* Earlier trainers may not have this middle position. Use HIGH position.

TABLE 12(e)

TEST POINT (Figure 8-1B)	SET SWITCHES G H J (INSTRUCTION)	SKETCH WAVEFORM	CHECK Figure 8-5
Flip-Flop #8 $\overline{Q}$ (T1)	1 1 0 (LOAD)		"INST" ① Check___
Flip-Flop #8 Q (T2)	SAME AS ABOVE		"OP" ② Check___
NOR #1 Output (T3)	SAME AS ABOVE		"BIT 0" ③ Check___
INSTRUCTION Reg. SHIFT PULSES (T4)	SAME AS ABOVE		④ Check___
ARITHMETIC Reg. SHIFT and BIT DELAY. T (T5)	SAME AS ABOVE		⑤ Check___
ARITHMETIC Reg. SHIFT and BIT DELAY T (T5)	0 1 1 (SHIFT RIGHT)		⑥ Check___
PROGRAM CTR. SET (T6)	0 1 1 (NO TRANSFER)		⑦ Check___
PROGRAM CTR. SET (T6)	0 0 1 (TRANSFER)		⑧ Check___

TEST POINT (Figure 8-1B)	SET SWITCHES G H J (INSTRUCTION)	SKETCH WAVEFORM	CHECK Figure 8-5
PROGRAM CTR SET (T6)	0 0 0 (TRANSFER ON NEGATIVE) (Ground A Reg. DATA IN)*		(9) NO NEGATIVE NUMBER Check____
PROGRAM CTR SET (T6)	0 0 0 (TRANSFER ON NEGATIVE)(Remove Ground to A Reg. DATA IN)*		(10) NEGATIVE NUMBER Check____
MEMORY READ/WRITE (T7)	1 1 0 (LOAD)		(11) Check____
MEMORY READ/WRITE (T7)	0 1 0 (STORE)		(12) Check____
BIT DELAY NS (T8)	1 0 1 (ADD)		(13) Check____
BIT DELAY NS (T8)	1 0 0 (SUBTRACT)		(14) Check____
NOT #1 Output (T9)	1 0 1 (ADD)		(15) Check____
NOT #1 Output (T9)	1 0 0 (SUBTRACT)		(16) Check____
FULL ADDER EA (T10)	0 1 0 (STORE)		(17) Check____

* When DATA IN on A. REG. is not grounded the A. REG. will contain all "1"s and the TRANSFER ON NEGATIVE instruction will detect a NEGATIVE NUMBER.

TEST POINT (Figure 8-1B)	SET SWITCHES G H J (INSTRUCTION)	SKETCH WAVEFORM	CHECK Figure 8-5
FULL ADDER EA (T10)	1 1 0 (LOAD)		(18) Check____
OBSERVE L3 and L4	1 1 1 (HALT)	COMPUTER TURNS OFF L3 = ON L4 = OFF	NONE Check____

The observed waveforms should be the same as the waveforms given in figure 8-5. When the waveforms are correct proceed to Experiment No. 9.

OBJECTIVE:

This experiment completes the construction of the Automatic Digital Computer. The Arithmetic and Memory Units are wired using the diagrams furnished in Experiment No. 8. Test and Maintenance Programs to verify operation and aid in Logic Troubleshooting are also given.

DESCRIPTION:

The construction of the computer design presented in the previous experiment is completed in this experiment.

STEP 2: ARITHMETIC UNIT:

The Arithmetic & Memory portion of the computer is shown in Figure 8-1C in the previous experiment. This experiment adds the wiring for the Memory, Program Counter, Instruction Register, Arithmetic Register, Adder, Bit Delay, and Exclusive-OR for Subtraction.

The detailed Instruction Table is given below. A convenient short-form Instruction Table is presented on page 3 in this experiment. The MNEMONIC coding is a shorthand instruction notation.

INSTRUCTIONS				
CODE		INSTRUCTION		Detailed Description
BINARY	OCTAL	NAME	MNEMONIC	
0 0 0	0	TRANSFER ON NEGATIVE	TON	The address from the Instruction Register is transferred to the Program Counter when the Arithmetic Register Bit 7 is a "1". The program counter steps by "1" after this instruction. This command could be called "TRANSFER PROGRAM ON BIT A7=1 TO ADDRESS PLUS 1"
0 0 1	1	TRANSFER	TRA	The address from the Instruction is transferred to the Program Counter. The counter then steps by 1. The command is "TRANSFER TO ADDRESS PLUS 1"

INSTRUCTIONS (CONT.)				
CODE		INSTRUCTION		
BINARY	OCTAL	NAME	MNEMONIC	
0 1 0	2	STORE (WRITE IN MEMORY)	STO	The data in the A Register is stored in memory at the address given in the instruction. The data also remains in the Arithmetic Register.
0 1 1	3	SHIFT RIGHT ONE	SHR	The data in the Arithmetic Register (Accumulator) is shift right once (END AROUND). The least significant bit (BIT 0) becomes the most significant bit (BIT 7).
1 0 0	4	SUBTRACT	SUB	The data in the memory at the instruction address location is subtracted from the number in the Accumulator. The memory data remains in the memory. Negative numbers appear in 2's complement form.
1 0 1	5	ADD	ADD	The data in the memory at the instruction address location is added to the contents of the Accumulator. The result is in the Accumulator. The memory data remains in the memory.
1 1 0	6	LOAD	LOD	The data in the memory at the instruction address location is loaded into the Accumulator. The previous Accumulator contents are cleared. The memory data remains in the memory. The previous Accumulator contents are lost.

INSTRUCTIONS (CONT.)				
CODE		INSTRUCTION		DETAILED DESCRIPTION
BINARY	OCTAL	NAME	MNEMONIC	
1 1 1	7	HALT	HLT	Halts the operation. Control goes to OFF state. Program Counter goes to next ADDRESS. Accumulator has added to it the number in memory corresponding to the instruction address. This command could be called "ADD and HALT".

A convenient short-form INSTRUCTION TABLE summary is given below:

INSTRUCTION TABLE		
INSTRUCTION	MNEMONIC	OCTAL
TRANSFER (+1) ON NEG	TON	0
TRANSFER (+1)	TRA	1
STORE	STO	2
SHIFT RIGHT (ONE)	SHR	3
SUBTRACT	SUB	4
ADD	ADD	5
LOAD	LOD	6
HALT (ADD)	HLT	7

PROCEDURE:

1. Wire the remaining logic for the Automatic Digital Computer given in Figure 8-1C in Experiment No. 8.

2. Be sure to connect the Memory inputs 2, 1, 0, to the Bit Counter B4, B2, B1 signals from FF#3Q, FF#4Q and FF#5Q.

3. Be sure to connect the leads to the terminals designated. Note that the A Register "S" inputs are really on-top on the panel, but on bottom in the logic diagram. Note that A, B, C on the Digital Decoder are drawn on the diagram in reverse from the labeling on the panel.

4. Be sure to connect the SELECT = 0 lead on the MEMORY to Ground. Be sure to connect the SELECT P=0 on the MULTIPLEXER to AND #1.

5. Be sure to connect all the Decoder outputs N0, N1, N2, N3, N4, N6, N7. There is no connection to Decode output N5. Be sure to connect the Decoder input EN to INST. Be sure to connect N7 from the Arithmetic Register to NOR #2. Note that EXCLUSIVE OR #1 and #2 are not used. Be sure to connect ground between the two units, as shown in the logic. You can check each logic gate for an output connection lead and for the proper number of inputs as tabulated below:

LOGIC		INPUTS
AND	1	2
	2	2
	3	2
	4	3
NAND	1	4
	2	2
	3	3
NOR	1	4
	2	4
	3	4
OR	1	4
	2	4
	3	3
EX OR	1	(not used)
	2	(not used)
	3	2
	4	1
NOT	1	1
	2	1
	3	1
	4	1
	5	1

You can check to see that all flip-flops have inputs on the required terminals as tabulated below:

FLIP-FLOP	TERMINALS USED				
1	T	J	K	Q	$\overline{Q}$
2	T	J	K	Q	$\overline{Q}$
3	T	Q			
4	T	Q			
5	T	Q			
6	$\overline{R}$	$\overline{S}$	Q		
7	T	$\overline{R}$	Q	$\overline{Q}$	
8	T	$\overline{R}$	Q	$\overline{Q}$	
9	T	$\overline{S}$	Q	$\overline{Q}$	
10	T	Q			

When the wiring is complete, you can proceed to test the operation of the computer.

6. With Switch F DOWN, the Switches G, H, J, control the INSTRUCTION CODE. Set Switch F DOWN to test Data Entry.

7. Data Entry Test.

7.1 Set Switch F DOWN for Manual Command.

7.2 Set Switches G H J to 0 1 0 for STORE.

7.3 Enter a number in the A Register (1 2 3) as follows:

7.4 Depress KEY 8 to CLEAR the A Register.
(The Register should be 000)

7.5 Depress KEY 1:
(The Register should be 001)

7.6 Shift "P Button" (on the Ed-Lab 900) 5 times.
(The Register should be 010)

7.7 Depress KEY 2:
(The Register should be 012)

7.8 Shift "P Button" 5 times.
(The Register should be 120)

7.9 Depress KEY 3:
(The Register should be 123)

7.10 The Data Entry using the Keyboard and A Register, and Adder is operating properly.

8. P Counter Test

8.1 Keep Switch F DOWN.

8.2 Keep Switches G H J at 0 1 0 for STORE.

8.3 Set Switch K DOWN then up to CLEAR the Program Counter. (The Address-Display Lamps on Ed-Lab 900 should be 0 0 0 0 0)

8.4 Set Switch E on the Ed-Lab 700 to one-cycle; DOWN.

8.5 Press the START Button, M on the Ed-Lab 700. (The Address should count to 0 0 0 0 1)

8.6 Keep pressing the M Button and the address should count in binary.

The P Counter Test is complete.

9. Memory Test

You will test writing (STORE) and reading (LOAD) from MEMORY.

9.1 Keep Switch F (900) DOWN for manual operation control.

9.2 Keep Switches G H J (900) at 0 1 0 for STORE.

9.3 Set Switch K (900) DOWN then UP to CLEAR the Address Counter to 0 0 0 0 0.

9.4 Set Switch E (700) DOWN for ONE CYCLE.

9.5 Using the Keyboard (900) and the P Switch (900) (5 times) enter the Octal number 2 1 6.

9.6 Depress the START BUTTON M (700).
The Data should be entered into memory in one INSTRUCTION/ OPERATE Cycle.
The Data should be retained in the display and A Register.
The Program Counter should be automatically stepped to binary 0 0 0 0 1 (Display Lamps on 900) which in Octal is 0 1.

9.7 Continue to enter the following six numbers into memory. Use Key 8 to Clear the Register as before. Use Keyboard, P Switch 5 times, and M Switch to enter data into memory.

MEMORY ADDRESS DISPLAY LAMPS (900)	DATA	CHECK
0 0	216	
0 1	047	
0 2	151	
0 3	236	
0 4	362	
0 5	003	

9.8 To Read the Data out of memory, first set Switches G H J to 110 for (LOAD).

9.9 Set Switch K (900) DOWN then UP to CLEAR Address to 0 0.

9.10 Leave Switch F (900) DOWN for Manual operation. Leave Switch E (700) DOWN for ONE CYCLE.

9.11 Press the M Button (700) to Readout the data from memory address 0 0. The Display should now be 216. Check the box in the above table if this is correct.
The Address should switch to 0 1 (octal).

9.12 Continue to Press the M Button and each data word will be read-out in sequence and the address will change to the address for the next data. The address lamps always indicate the address location one word higher than the data being displayed. Check-off each word as it is read-out properly.

9.13 You can select Data Words for all 32 data locations for address 0 0 to 37 to test the complete memory. Enter these numbers consisting of 3 octal digits in a table. Enter the numbers into memory. Then read-out the memory to check the data.

(The student may skip this full memory test and continue.)

10. Control Pulse Tests

It is possible to monitor and test all Control Pulses using an oscilloscope as follows. It is not necessary for the student to observe these waveforms this time.

10.1 Set Switch F DOWN for manual instruction.

10.2 Set Switch K DOWN: Counter at 0 0.

10.3 Set Switch E UP for AUTO.

10.4 Set Switches G H J for 0 1 0 STORE.

10.5 Press M Button to START.

10.6 View all Pulse Waveforms on an oscilloscope.
Set in INSTRUCTION Codes on G H J to view waveforms.

WRITE	(NAND #3)	(Pulses only on STORE 0 1 0)
NOTE: SYNC ON OPERATE SIGNAL, LAMP 2 of Ed-Lab 700.		
TRANSFER	(NAND #1)	(Pulses only on TRANSFER)
SHIFT A	(OR #2)	(Pulses on OP cycle)
SHIFT I	(AND #3)	(Pulses on INST cycle)
OP	(FF#8Q)	
INST	(FF #8 $\overline{Q}$)	
BIT 0	(NOR #1)	

11. Computer Operation Instruction Tests

The Computer Operation Test is performed by running programs using each instruction. These are called DIAGNOSTIC PROGRAMS or MAINTENANCE PROGRAMS and are used in verifying proper computer operation and to provide information when troubleshooting.

The LOAD tests program is shown below:

1. LOAD TEST (INSTRUCTION #6)

PROGRAM

(1) MEMORY LOCATION	(2) WORD	(3) ACCUMULATOR CONTENTS
00	03(6)	113
01	04(6)	345
02	05(6)	267
03	113	
04	345	
05	267	

The first step is to store the 6 words from column 2 in memory locations 00 through 05. These words are then "read-out" to verify proper entry into memory. (The parenthesis around 6, i.e. (6), is only for convenience to indicate an instruction.) The computer program counter is then reset (Switch K DOWN then UP) to 00. The computer control is returned to the instruction-- Switch F UP. Use the manual cycle (Switch E DOWN). The computer is now ready for the first instruction (00). Depress M, and the instruction 036 is carried out with the resultant 113 in the accumulator and on the display. At this time, the program counter is stepped to 01. The Accumulator Contents, column 3, shows the display after the step from column 1 is performed. The test program is run in three steps; 00, 01 and 02.

The LOAD TEST program is shown below:

	1. LOAD TEST (6)		ACCUMULATOR
START →	00	03(6)	113
	01	04(6)	345
STOP →	02	05(6)	267
	03	113	
	04	345	
	05	267	

The LOAD TEST procedure is:

(1) ENTER DATA 00 to 05 (Check data entry)

(2) START AT 00 (Switch K DOWN then UP) (Switch F UP, Switch E DOWN)

(3) MANUAL CYCLE 3 TIMES (Depress M)

(4) DISPLAYS 113
345
267

Run the LOAD TEST.
Then run the other tests as given below:

2. ADD TEST (5) ACCUMULATOR

START →	00	03 (6)	113	LOAD
	01	04 (5)	260	
STOP	02	05 (5)	262	
	03	113		
	04	145		
	05	002		

(1) ENTER DATA 00 to 05

(2) START AT 00

(3) MANUAL CYCLE 3 TIMES

(4) DISPLAYS 113
260
262

	3. SUBTRACT TEST (4)		ACCUMULATOR
START	00	03 (6)	113
	01	04 (4)	031
STOP →	02	05 (4)	027
	03	113	
	04	062	
	05	002	

	4. STORE TEST (2)		ACCUMULATOR
START	00	04 (6)	345
	01	06 (2)	345
	02	05 (6)	267
STOP →	03	06 (6)	345
	04	345	
	05	267	

	5. SHIFT RIGHT (3)		ACCUMULATOR
START	00	04 (6)	345
	01	00 (3)	362
STOP →	02	05 (3)	171
	03	113	
	04	345	
	05	267	

	6. TRANSFER (1)		ACCUMULATOR
START	00	04 (6)	345
	01	02 (1)	345
	02	01 (6)	↓ SKIPS
STOP →	03	05 (6)	267
	04	345	
	05	267	

7. TRANSFER ON NEG (0) ACCUMULATOR

START	00	06 (6)	145	(Load Positive number)
	01	04 (0)	145	(Positive, so no transfer)
	02	05 (6)	267	(Load NEG number)
	03	06 (0)	267	(transfers to 07)
	04	06 (6)		
	05	267		SKIPS
	06	145		
STOP	07	02 (6)	056	

8. HALT (7) ACCUMULATOR

(Set Switch E UP AUTO)

START	00	03 (6)	113	
	01	04 (5)	170	
	02	05 (7)	203	STOPS AUTOMATICALLY program counter halts at 03.
	03	113		
	04	055		
	05	013		

The above HALT instruction is a HALT and ADD.
One single program can test all instructions for computer check-out as given on the following page.

12. Maintenance Test Program

This program tests all Instructions. It should always be used after wiring the computer to verify operation.

	MEMORY		ACCUMULATOR	TEST	
START	00	21 (6)	013	LOAD	(6)
	01	22 (5)	360	ADD	(5)
	02	23 (4)	071	SUB	(4)
	03	25 (2)	071	STORE	
	04	21 (6)	013		
	05	25 (6)	071	STORE	(2)
	06	24 (5)	072	ADD	
	07	00 (3)	035	SHIFT RIGHT	(3)
	10	11 (1)	035	TRANSFER JUMP	
	11	21 (5)	↓ SKIP	JUMP	
	12	23 (5)	324	(TRANSFER)	(1)
	13	14 (0)	324	TRANSF NEG JUMP	(-)
	14	24 (5)	↓ SKIP		
	15	11 (4)	107		
	16	11 (0)	107	TRANSF NEG	(0)
	17	21 (5)	122		
STOP →	20	21 (7)	135	HALT	(7)
	21	013			
	22	345			
	23	267			
	24	001			

Enter the above Maintenance Test Program into memory. In the AUTOMATIC mode, the computer should start at 0 0 and automatically HALT (OFF) at Program Counter 21 with a Display Output of 135. In the MANUAL step mode, the program should step as shown and SKIP Program Counter steps 11 and 14 (octal).

OBJECTIVE:

This experiment describes the techniques of computer programming, symbolic programs, program coding, and flow charts and provides many examples and exercises to be used with the AUTOMATIC DIGITAL COMPUTER constructed in the previous experiments.

DESCRIPTION:

The PROGRAM is the sequence of instructions, stored in the computer memory, which directs the computer operation in alternating Instruction and Operate Cycles to solve a given problem. Typical programs were presented earlier and the program for Addition is repeated below for X = A + B.

MEMORY LOCATION	INSTRUCTION ADDRESS	INSTRUCTION COMMAND	ACCUMULATOR CONTENTS
00	17	LOAD	A
01	25	ADD	A + B (=X)
02	24	STORE	X
03	14	HALT	X

The operation consists of loading "A" from memory location 17 into the Accumulator; adding "B" from memory location 25 (resulting) in X; storing the result (X) in memory location 24; and Halting the operation. The computer starts with the first instruction at memory location 00 in the above program. Instructions are stored in locations 00, 01, 02, and 03. Input Data is stored in locations 17 and 25. Output Data is stored in location 24. The Address (location 14) used in the HALT instruction is arbitrary. Having a "000" in that address (14) leaves the accumulator unchanged. The complete program is written as shown below:

LOCATION	INSTRUCTION
00	17 LOAD
01	25 ADD
02	24 STORE
03	14 HALT
14	000
17 ϕ	A
24 ϕ	X
25 ϕ	B

"ϕ" denotes here that this location is used later in the program and can be left "empty" in the program.

The $\emptyset$ (eg. 17 $\emptyset$) indicates input or output numbers and it is not necessary to enter that data for the general program.

"Operator Instructions" for use of the program are:

1. ENTER A in 17
2. ENTER B in 25
3. ENTER 000 in 14
4. START AT 00 LOCATION
5. ANSWER X is in 24
   ANSWER X ON DISPLAY

When initially preparing the program, it is not necessary to use the exact memory location. A short-hand or "Symbolic Program" is written. Symbols or letters are used to specify the addresses rather than the exact binary or octal location numbers.

The letter "L" denotes "location". L1 refers to an arbitrary memory Location (address). L2 refers to another arbitrary Location; LA refers to the Location of A; LNO is the Location of Number 0; etc. The program is then written as:

SYMBOLIC PROGRAM

PROGRAM X = A + B	
LOCATION	WORD
L1	LA LOAD LB ADD LX STORE LNO HALT
LNO	000
LA - $\emptyset$	A
LB - $\emptyset$	B
LX - $\emptyset$	X

The location address does not have to be written when a memory location is directly after the previous location.

A subtract SYMBOLIC PROGRAM would be written as:

SUBTRACT Y = A - B	
L2	LA LOAD
	LB SUB
	LY STORE
	LNO HALT
LNO	000
LA - Ø	A
LB - Ø	B
LY - Ø	Y

Another subtract program is:

SUBTRACT Z = B - A	
L3	LB LOAD
	LA SUB
	LZ STORE
	LNO HALT
LNO	000
LA - Ø	A
LB - Ø	B
LZ - Ø	Z

The SYMBOLIC PROGRAM is also called an ASSEMBLY PROGRAM. The CODED program using the computer machine addresses and codes is called the MACHINE LANGUAGE PROGRAM or the CODED PROGRAM.

Another Program is:

W = 2A - B	
L4	LA LOAD
	LA ADD
	LB SUB
	LW STORE
	LNO HALT
LNO	000
LA - Ø	A
LB - Ø	B
LW - Ø	W

Consider the composite program to solve all three (W, X, Y, Z):

X = A + B Y = A - B Z = B - A W = 2A - B	
L1	LA LOAD
	LB ADD
	LX STORE
	LNO HALT
L 2	LA LOAD
	LB SUB
	LY STORE
	LNO HALT
L3	LB LOAD
	LA SUB
	LZ STORE
	LNO HALT
L4	LA LOAD
	LA ADD
	LB SUB
	LW STORE
	LNO HALT
LNO	000
LA - Ø	
LB - Ø	
LW -Ø	
LX - Ø	
LY - Ø	
LZ - Ø	

The Operator Instructions are then:

OPERATION INSTRUCTIONS

PROGRAM NAME: M = F (A, B) = Function of A, B		
LOAD		NOTES
LOCATION	WORD	
00	__ (1)	TRANSFER First 2 digits 04 for A + B* 10 for A - B 14 for B - A 20 for 2A- B
01	___	LOAD A (Note: Transfer Ins
02	___	LOAD B
03	Ø	Start at 00 Answer in 03 and in Accumulator Display

*NOTE: Transfer Instruction actually transfers to address +1; see Experiment 9, page 1.

The operator is thus instructed to load the memory word in location 00 with the function desired; to load 01 and 02 with A and B respectively; and the result will appear on the display and in location 03.

PROCEDURE:

Enter the above program (M = F (A, B)) in memory. Read - out the memory and check data. Select different values for A and B and solve for each of the functions. Enter the results in the table below. Select numbers which do not result in an "overflow" as described in a previous experiment.

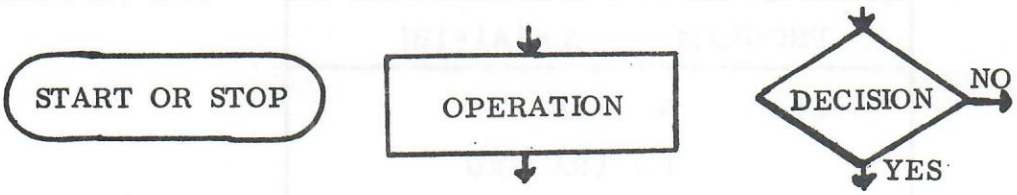
PROGRAM TEST

PROBLEM	A	B	FUNCTION	COMPUTER ANSWER	CHECK
1	016	034	A + B		
2			A - B		
3			B - A		
4			2A - B		
5					
6					
7					

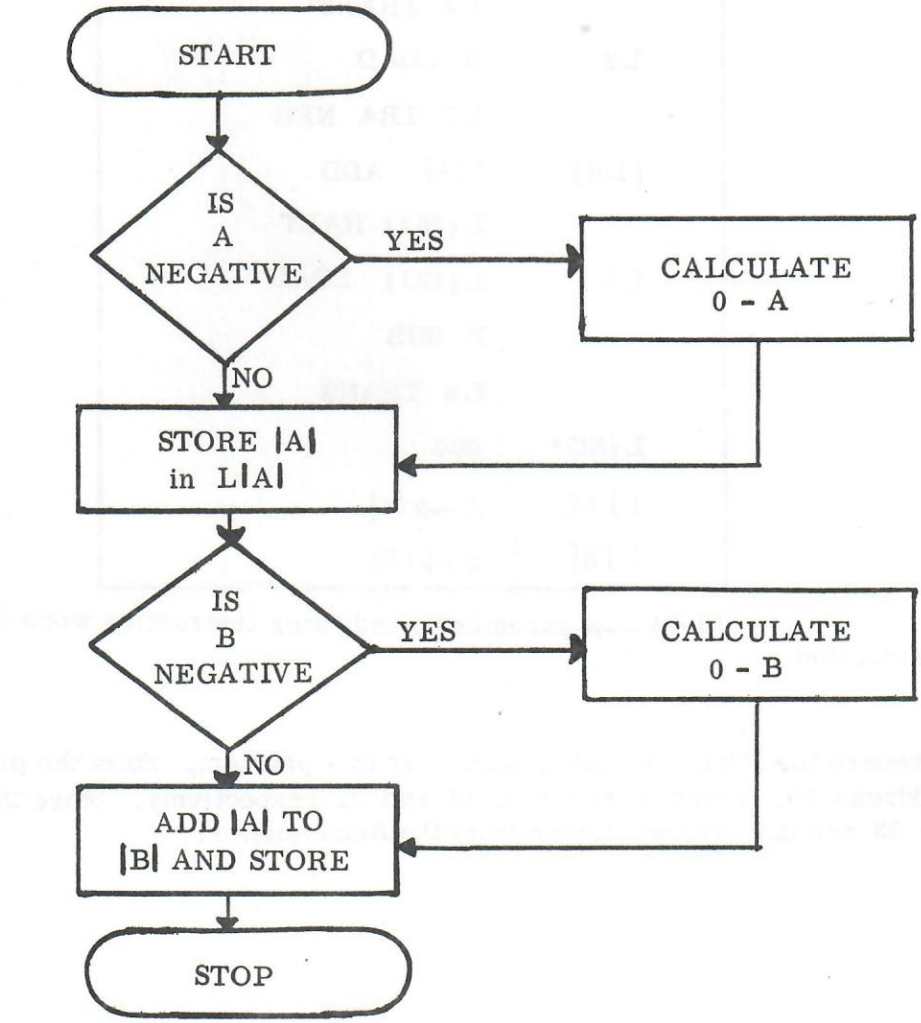
Use both positive and negative numbers coded in "2's complement octal".

DESCRIPTION (Flow Charts):

A Flow Chart is a graphical representation of program operation and is usually the first step in a complex program. The following graphical symbols are used:



Consider the problem of solving for the sum of the magnitude of two numbers; i.e. $X = |A| + |B|$ (eg. if $A = 5$, $B = -3$; then $X = 5 + 3 = 8$). The flow chart is:



FLOW CHART $X = |A| + |B|$

The symbolic program is written as

SYMBOLIC PROGRAM

PROGRAM	X = |A| + |B|
L1	A LOAD
	L2 TRA NEG
(L4)	L|A| STORE
	L3 TRANS
L2	L(NO) LOAD
	A SUB
	L4 TRANS
L3	B LOAD
	L5 TRA NEG
(L6)	L|A| ADD
	L(NO) HALT
L5	L(NO) LOAD
	B SUB
	L6 TRANS
L(NO)	000
L|A|	A → |A|
L|B|	B → |B|

(L4) → parenthesis indicates instruction word follows previous instruction.

Prepare the Program Coding Chart for this problem. Start the program at address 00. Enter A and B at 01 and 02 respectively. Store the answer X in 03 and display the answer from the Accumulator.

CODING

X = |A| + |B|		
ADDRESS	WORD	COMMENTS
0 0	0 4 1	TRA to 05
0 1	ø	A
0 2	ø	B
0 3	ø	X
0 4	0 3 6	LOAD X
0 5	0 1 6	LOAD A
0 6	1 0 0	TON to 11
0 7	2 5 2	STORE |A| in 25
1 0	1 3 1	TRA to 14
1 1	2 4 6	LOAD 0 0 0
1 2	0 1 4	SUB A results in |A|
1 3	0 6 1	TRA to 07
1 4	0 2 6	LOAD B
1 5	2 0 0	TON to 21
1 6	2 5 5	ADD |A| from 25
1 7	0 3 2	STOR X in 03
2 0	2 4 7	HALT (ADD contents of 24=000)
2 1	2 4 6	LOAD 0 0 0
2 2	0 2 4	SUB B results in |B|
2 3	1 5 1	TRA to 16
2 4	0 0 0	ZERO
2 5		(STORAGE for PARTIAL SUM)
2 6		
2 7		
3 0		
3 1		
3 2		
3 3		
3 4		
3 5		
3 6		
3 7		

Prepare the Operator Instructions

OPERATOR INSTRUCTIONS		$X = \|A\| + \|B\|$
LOAD (041)	in	ADDRESS 0 0
LOAD A	in	0 1
LOAD B	in	0 2
START at 0 0		
AUTOMATIC HALT		
ANSWER IN MEMORY 03 and DISPLAY		

Enter the program in memory. Read - out and check the memory. Select various inputs and run the program.

PROGRAM TEST

PROGRAM X = |A| + |B|				
PROBLEM	A	B	X	CHECKED
1	005	002	007	
2	-005 (373) *	-002 (376)*	007	
3	010	-012 (366)*	022	
4	- 020 (360)*	010	030	

* ENTER NEGATIVE NUMBER (2's COMPLEMENT)

QUESTIONS:

1. Program the solution for $Y = |2A - B|$.
   (a) Prepare the Flow Chart.
   (b) Prepare the Symbolic Program (or Assembly Program).
   (c) Prepare the Coding Chart (in Machine Language).
   (d) Prepare the Operator Instructions.
   (e) Prepare a Program Test Sheet.

OBJECTIVE:

This experiment presents programs which use a continuous "loop" in order to solve problems such as MULTIPLICATION and DIVISION.

DESCRIPTION:

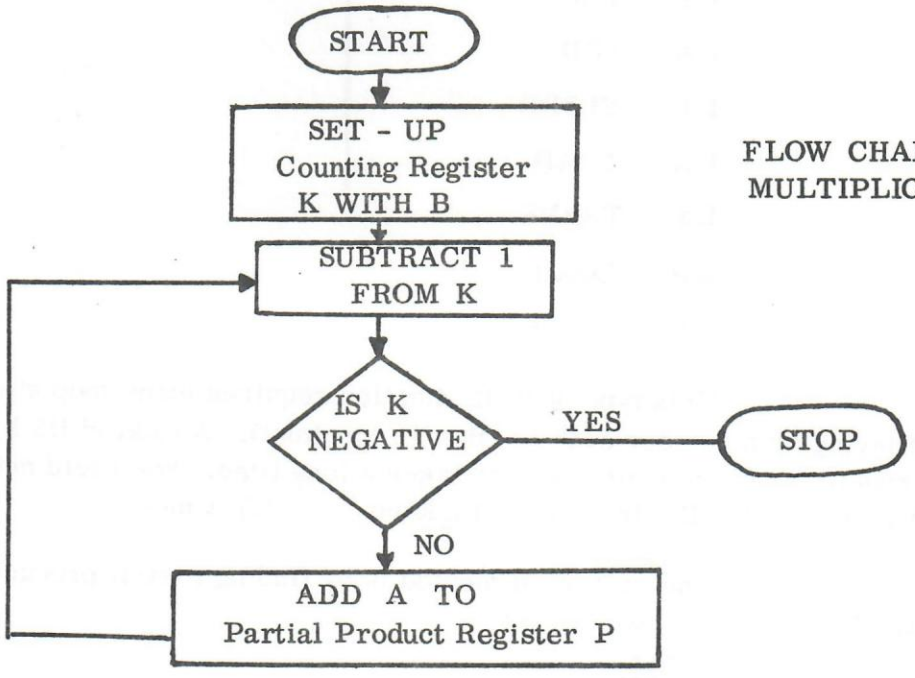
Consider the program for the multiplication problem;

$$X = A \bullet B$$

One method of multiplication is by continuous addition. Thus: 17 x 6 (decimal) numbers can be performed by summing 17 six times or:

$$\begin{array}{r} 17 \\ 17 \\ 17 \\ 17 \\ 17 \\ \underline{17} \\ 102 \end{array}$$

The flow chart is given below for this process of multiplication of $X = A \bullet B$ where A is summed B times. The number "K" keeps track of the number of times A is added. Number P sums the partial product.



FLOW CHART
MULTIPLICATION

The flow chart indicates a LOOP program operation.

The symbolic program is :

PROGRAM X = A • B

LOCATION	INSTRUCTION	
L(00) Ø	L1 TRANSFER	
LA Ø	A	
LB Ø	B	
LP Ø	X	
L(N0)	000	number 0
L(N1)	001	number 1
LK Ø	K	
L1	L(NO) LOAD	start program
	LP STORE	
	LB LOAD	
L3	L(N1) SUB	
	LK STORE	
	L2 TRANS NEG	
	LP LOAD	
	LA ADD	
	LP STORE	
	LK LOAD	
	L3 TRANS	
L2	LP LOAD	
	LNO HALT	

This type of multiplication requires many loop steps. Consider multiplying by a number such as B = 115 (decimal). A total of 115 loop operations are required and the multiplication takes a long time. We would not consider multiplying 2 by 115 by means of adding 2 a 115 times.

The preferred method is by finding partial products for each digit. Thus:

```
    2
x 115
---
   10   (5 x 2)
    2   (1 x 2)
---
    2   (1 x 2)
---
  230
```

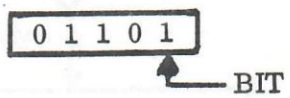
The same technique is used in binary numbers. Thus 8 x 13 is:

8	=	0 1 0 0 0	= A
x 13	=	0 1 1 0 1	= B
		0 1 0 0 0	
		0 1 0 0 0	
		0 1 0 0 0	
104	=	0 1 1 0 1 0 0 0	

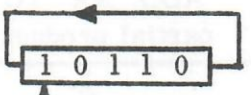
The program process is then to examine each bit of B, starting with the least significant bit (to the right), and adding A to a partial product. When looking at the third bit (from the left), we must add 4A, and so on. This is done 8 times for each bit of B.

Each bit of B is examined by using the "transfer on negative" instruction and by "shifting to the right".

Thus, for:

B = 

To examine the BIT

1. SHIFT B TO RIGHT 

2. TRANS ON NEG for a "1".

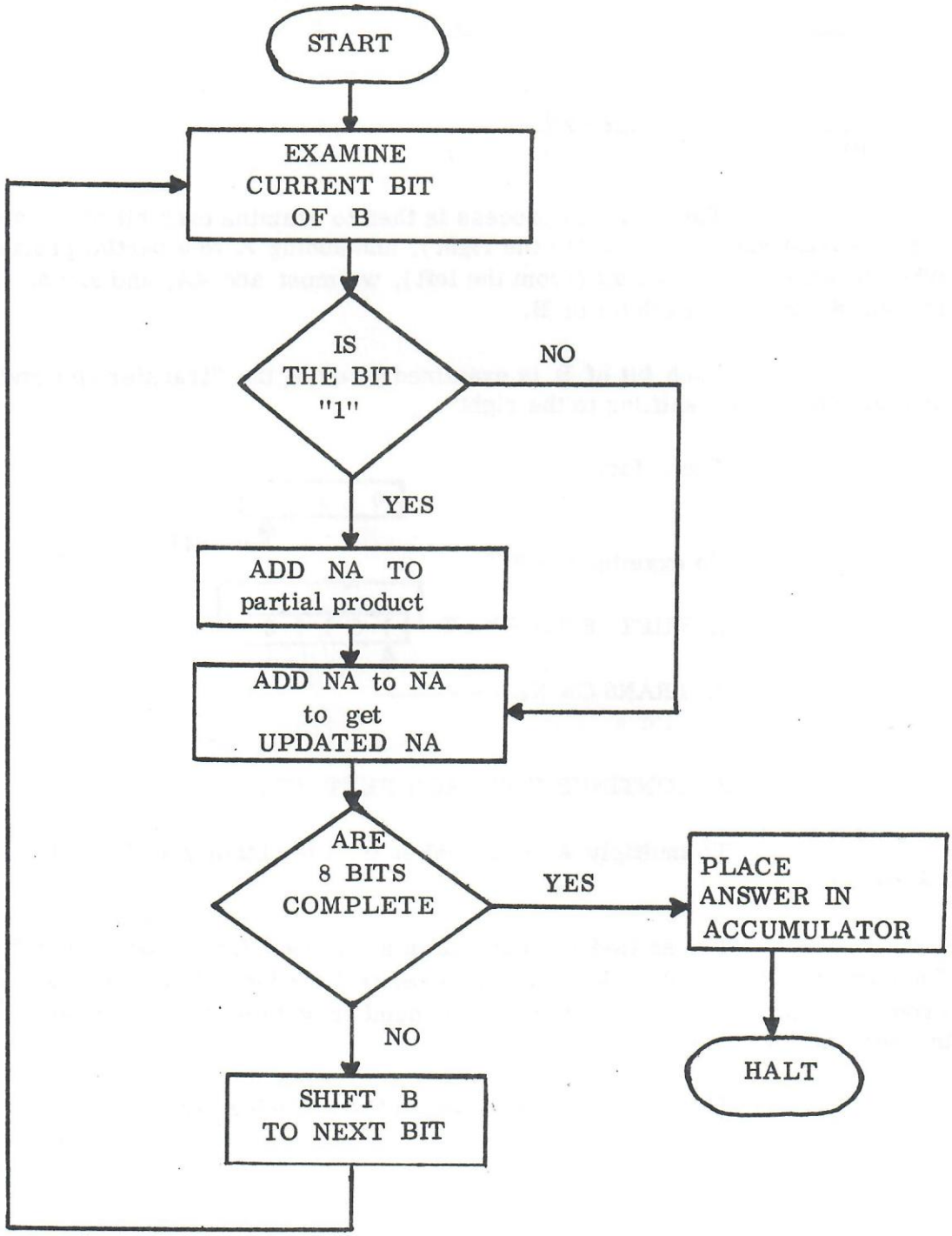
3. CONTINUE FOR EACH NEXT BIT.

To multiply A by 2; add A to A to obtain 2A; for 4A - add 2A to 2A.

The shifted B is stored in a Location LSB (Location Shifted B). The current A (i.e. 2A, 4A, etc.) is stored in Location LNA. The Partial Product is stored in Location LPP. The number of bits of B examined is stored in Location LNB.

The flow chart is shown on the following page.

FLOW CHART $X = A \cdot B$

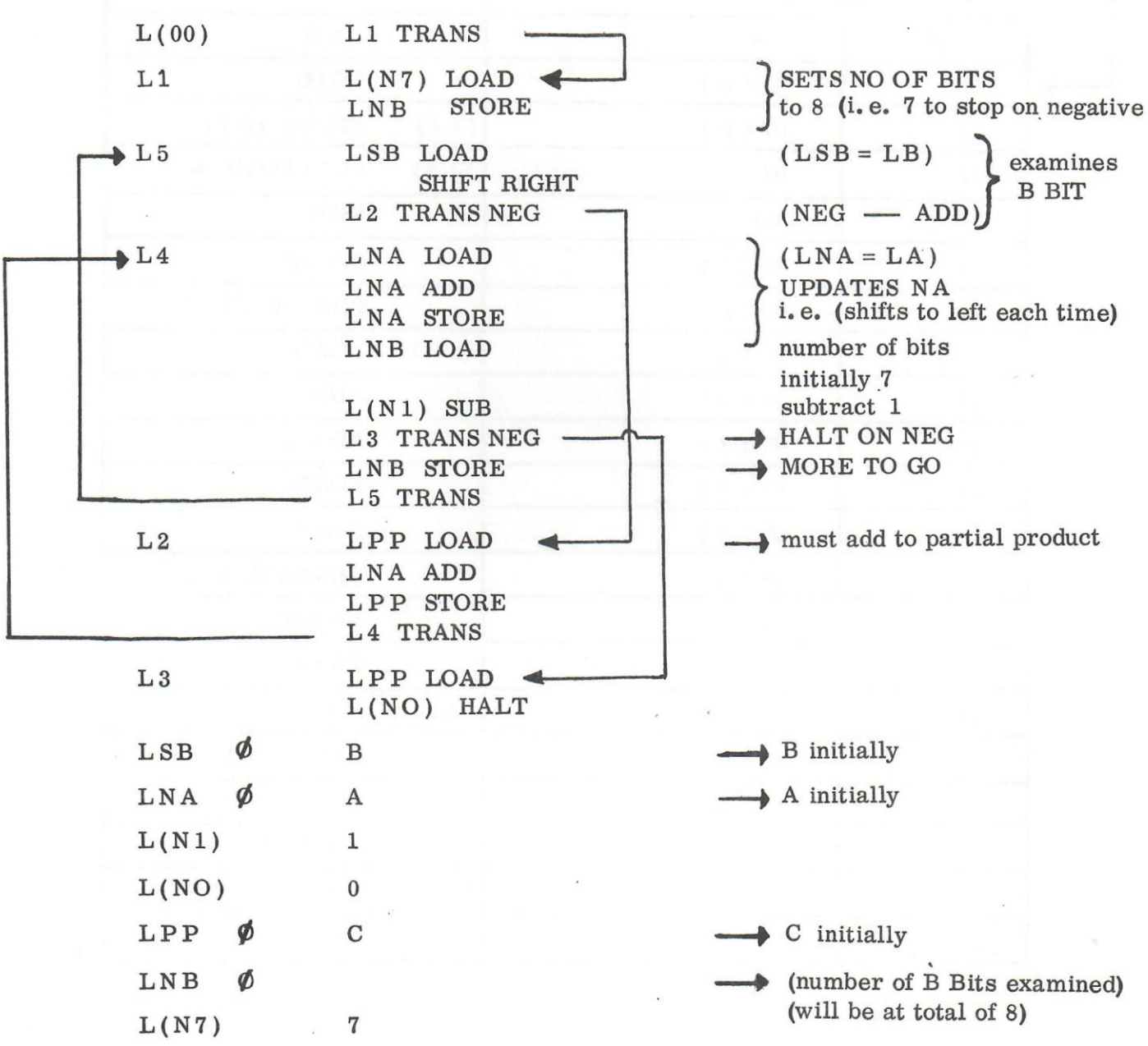


The flow chart leaves out some details. As an example; "ARE 8 BITS COMPLETE" is determined by subtracting "1" from a register (LNB) each time the loop is performed, and examining the sign bit for a negative number. Register LNB must initially be set up with 7 (to obtain a negative after 8 subtractions). Also, examining the current "bit of B" on the right hand side is performed by shifting around and examining the "sign" position.

The symbolic program is given below. By placing a number C initially in the partial product memory, the solution is obtained for

$$X = A \cdot B + C$$

SYMBOLIC PROGRAM $X = A \cdot B + C$



Coded Program X = A•B + C			
LOCATION	WORD		INST
00	07 (1)		TRANS (L1)
01 Ø	—		A (LNA)
02 Ø	—		B (LSB)
03 Ø	—		C (LPP)
04	000		L(NO)
05	001		L(N1)
06	007		LN7
07 Ø	—		LNB
10	06 (6)		LOAD
11	07 (2)	(L1)	STORE (N7)
12	02 (6)	(L5)	(L1) LOAD B
13	00 (3)		SHR
14	02 (2)		STORE
15	25 (0)		TON (L2)
16	01 (6)	(L4)	LOAD
17	01 (5)		ADD
20	01 (2)		STORE
21	07 (6)		LOAD
22	05 (4)		SUB 1
23	31 (0)		TRANS NEG L3
24	07 (2)		STORE
25	11 (1)		TRANS L5
26	03 (6)	(L2)	LOAD PP
27	01 (5)		ADD
30	03 (2)		STORE
31	15 (1)		TRANS (L4)
32	03 (6)	(L3)	LOAD PP
33	04 (7)		HALT

OPERATOR INSTRUCTIONS X = A•B + C
1 LOAD A in 01
2 LOAD B in 02
3 LOAD C in 03
4 START AT L(00)
5 ANSWER ON DISPLAY AND L(03)

TEST RESULTS					
TEST	A	B	C	EXPECTED ANSWER	ACTUAL ANSWER
1	010	007	003	073	
2	030	004	071	231	
3					
4					
5					

Consider test problem #1:

A = 010 (OCTAL) = 8 decimal

B = 007 (OCTAL) = 7 decimal

C = 003 (OCTAL) = 3 decimal

(ANSWER = 59 decimal)

Multiply in binary A = 00001000

B = 00000111

1000

1000

1000

PRODUCT 00111000

ADD C 00000011

ANSWER = 00111011 binary

ANSWER = 59 decimal (checks)

Use positive numbers for A, B and C on these problems. Check that answer X does not exceed octal 377. (This program will operate with negative A and C and maximum X and A•B ±177, (octal).

Considerations for OVERFLOWS, is presented in a later experiment.

Multiplication in octal (Test Problem #1):

		OCTAL	DECIMAL
A =	010	073 (8) =	0x64
B =	007		7x 8 = 56
product	070		3x 1 = 3
add C	003		59

073 octal = 59 decimal

Multiplication problem 2 (with octal carrys):

	OCTAL (8)	DECIMAL (10)
A =	030	24
xB =	004	x 4
	140	96
+C =	+ 071	+ 57
X =	231 (8)	153 (10)

OCTAL	DECIMAL	
231 (8) =	2 x 64 →	128
	3 x 8 →	24
	1 x 1 →	1
		153

CHECKS

PROCEDURE:

Enter the program into the computer. Read-out to check proper entry. Run various test problems and enter results in the table provided.

QUESTIONS:

1. Prepare the "Program" (i.e. Flow Chart, Symbolic Program, Operator Instructions, and Test Sheets) for the following problem:

"SEVEN NUMBERS ARE TO BE STORED IN LOCATIONS 01 THRU 07. FIND THE SUM OF ALL NUMBERS UNDER 25 (decimal). THE OPERATOR INSERTS THE 7 NUMBERS."

Note that in this problem, the loop sequential looks at the data in addresses 01 thru 07. We must thus change the address in an instruction word. This is called INSTRUCTION MODIFICATION.

OBJECTIVE:

This experiment demonstrates a number of interesting and useful techniques and applications of programming. These include full programming with very few available instruction commands, binary and binary coded decimal conversions, bit selections, overflow detection and subroutines. This experiment requires student programming and programs should be written prior to the computer laboratory periods.

DESCRIPTION:

The Ed - Lab 900 Computer System employs 8 different commands (HALT, ADD, STORE, etc.). Computers generally have many more commands such as 16, 24, 54, 96, different commands. The advantage of additional commands is in the reduction of instruction words required in memory. The Ed - Lab commands are more than adequate to solve any problems. Indeed, a number of these commands are not needed and the same functions could be accomplished using fewer commands.

Consider the problem:

$$X = A + B - C$$

The normal program steps are:

```
LOAD  A
ADD   B
SUBTRACT  C
STORE  X
```

Consider performing this problem without the ADD instructions. Perhaps you would like to try it before continuing to the solution on the next page.

Addition can be performed by subtracting a negative number. Thus

$$X = A - (-B) - C$$

and no addition is required. The program with the ADD is then:

```
LOAD  0
SUB   B
STORE (-B)
LOAD  A
SUB   (-B)
SUB   C
STORE X
```

It is thus seen that 7 instruction words are required rather than the 4 needed on the previous page with the ADD instruction.

LOAD is needed when any number (Y) is located in the Accumulator and we wish to LOAD A. If "0" were in the Accumulator, we could load A by ADD A.

A "0" accumulator is obtained by subtraction of the contents of the Accumulator from itself. Thus, for an Accumulator starting with Y, the program to load A, without a load instruction, is:

```
STORE Y
SUB   Y      (results in 0 ACCUMULATOR)
ADD   A
```

This uses three instructions rather than only one. (LOAD A).

PROCEDURE: (MINIMUM COMMANDS) :

Using the procedures demonstrated above, prepare the symbolic and coded programs to solve the problem:

$$X = A + B - C$$

Without using both the LOAD and the ADD operation, enter the program below:

PROGRAM FOR X = A + B - C

(WITHOUT LOAD OR ADD INSTRUCTIONS)

Enter the program in the computer and try some test problems. Enter the results below:

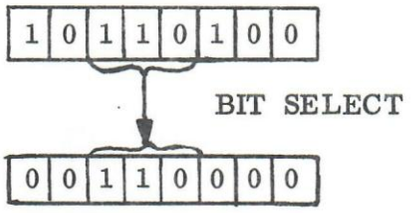
X = A + B - C

TEST	A	B	C	X
1	004	007	005	006
2				
3				
4				

If the proper answers are not calculated, then the program is in error and PROGRAM DE - BUGGING is to be performed. This can be accomplished by a step - by - step operation of the routine in Manual Cycle mode.

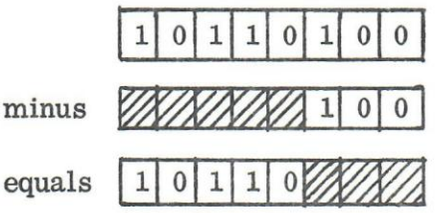
DESCRIPTION (BIT SELECTION):

Bit selection consists of removing certain data bits from a word as illustrated below:



The elimination of high order bits (left) is accomplishing by doubling the number. The binary number is shifted to the left, and the left most bit disappears. Thus, if we wish to remove the 2 left bits, we would multiply by 4, which requires 2 ADDITIONS (i.e. Y = A + A and 4A = Y + Y), and then shift right twice.

To eliminate the 3 right bits, we would first eliminate the left 5 bits as described above (5 Additions and 5 right shifts) and subtract this number from the original number. i.e.

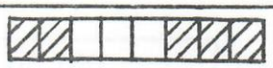


Removing the center bits is a combination of these two techniques.

PROCEDURE (BIT SELECTION):

Prepare the program for the removal of the 3 right bits and 2 left bits. This selects the center octal digit.

Enter the program in the computer and perform various test programs.

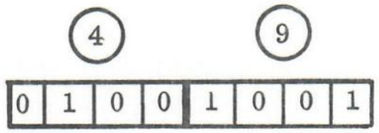
BIT SELECTION 		
TEST	INPUT	OUTPUT
1	136	030
2	264	060
3		
4		
5		

QUESTIONS:

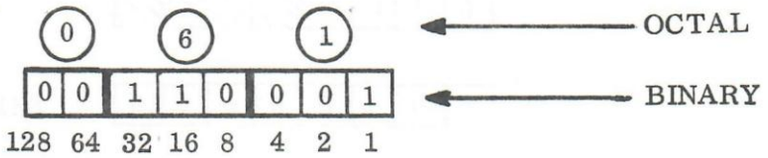
1. What are the program coding steps for the above bit select problem?

DESCRIPTION (BCD TO BINARY CONVERSION):

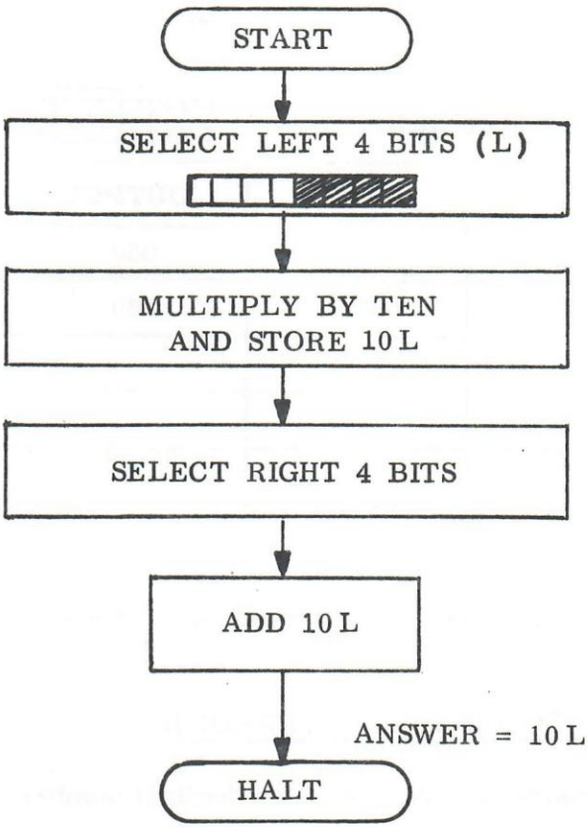
Consider the binary coded decimal number.



It is desired to convert this to the binary number for 49. The binary equivalent is:

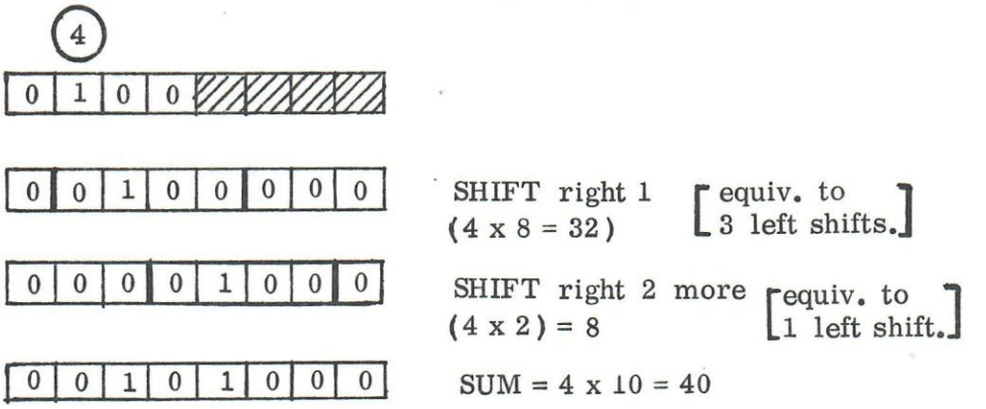


This is performed as shown in the following flow chart.



Multiplication by 10 is easily performed as a multiplication by (2 plus 8). Multiplication by 2 and 8 is performed by shifting.

Example:


4
0 1 0 0
0 0 1 0 0 0 0 0
SHIFT right 1
(4 x 8 = 32)
[equiv. to
3 left shifts.]
0 0 0 0 1 0 0 0
SHIFT right 2 more
(4 x 2) = 8
[equiv. to
1 left shift.]
0 0 1 0 1 0 0 0
SUM = 4 x 10 = 40

Then add the right hand 4 bits,

thus: 40 + 9 = 49

QUESTION:

2. Prepare the Program for BCD to binary conversion.

PROCEDURE:

Enter the program in the computer and run tests. (See Test 1 below: Data for BCD 49 is entered as octal 111.)

BCD TO BINARY		
TEST	BCD (CODE)	BINARY (OCTAL)
1	④ ⑨ 0100 1001	061
2		
3		
4		
5		

QUESTION:

3. Prepare the Symbolic Program for a Binary to BCD program. Try a written example in binary.

Hint:

The most significant BCD digit "tens" is obtained by subtracting ten and counting the number of times this can be done until a negative number is obtained. This gives the "tens" digit plus 1. The number (ten) is added back. The remainder is the units digit.

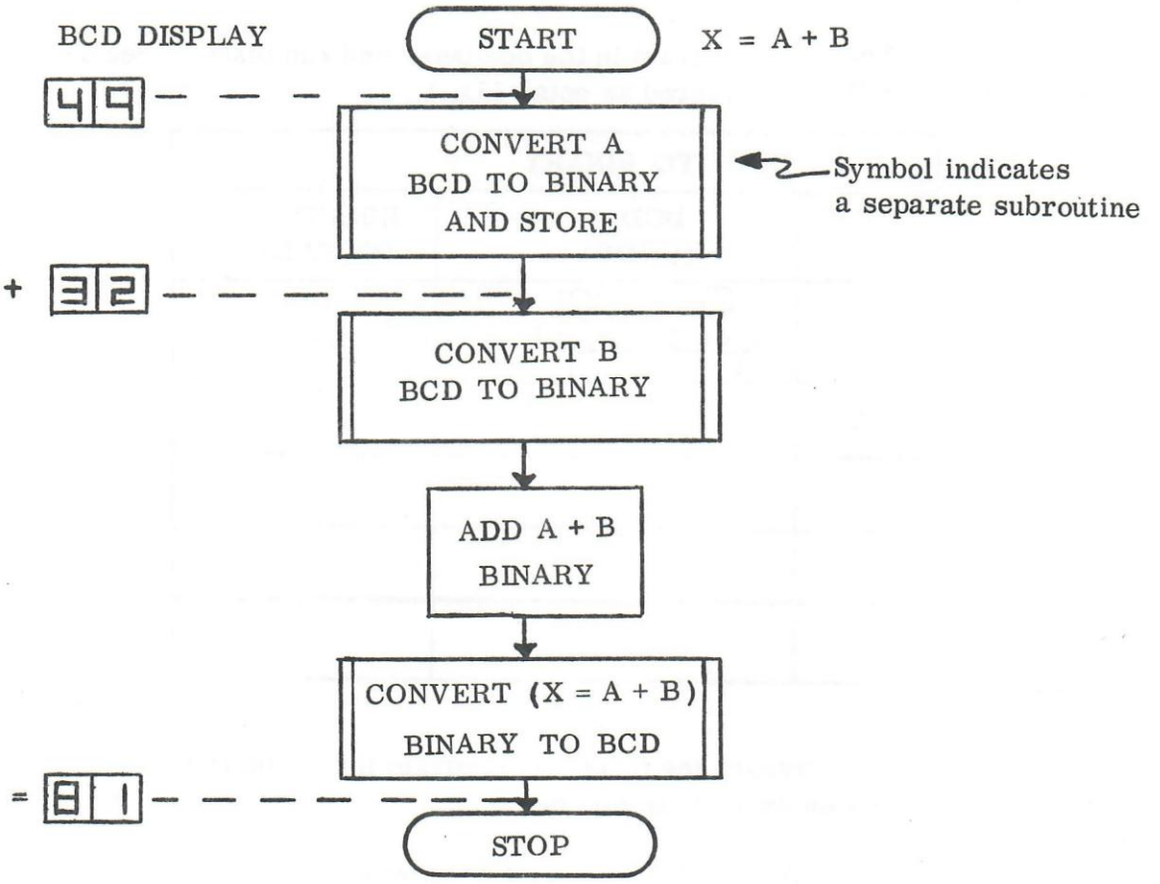
Example:

36	
-10	ONE TIME
26	
-10	TWO TIMES
16	
-10	THREE TIMES
6	
-10	FOUR TIMES
- 4	NEGATIVE
+10	BACK TO THREE TIMES
6	REMAINDER SIX

The BCD number is thus THREE SIX.

DESCRIPTION (DECIMAL COMPUTER):

Consider the computer with the BCD input keyboard and display configuration. The computer adds in binary. The addition program would thus be:



DESCRIPTION (OVERFLOW):

When adding two numbers, it is possible to have an overflow because the maximum octal number is 377. Computers generally look for an overflow using logic to detect an extra "carry" on the carry flip-flop (Bit Delay) after the completion of an operation. The computer is stopped on an overflow. When using positive and negative (complement) numbers, overflow detection is more complicated. In the programs we have used, our selection of numbers to be used in the problems was such as to avoid overflow. This is necessary because even if we had a Halt on Overflow feature, the problem that is programmed would not be solved.

Overflow will only occur on ADD or SUBTRACT. When a positive number and a negative number are added by the ADDER there can be no overflow. The ADDER only adds; subtraction is accomplished by complements. When two positive numbers are added and the sum is negative - There Is An Overflow. When two negative numbers are added and the sum is positive - There Is An Overflow.

The determination can be made using logic elements by hardware or by programming instructions (by software).

The Overflow Truth Table is:

Overflow Truth Table

A	B	Sum	Overflow
0	0	1	1
1	1	0	1
ALL OTHERS			0

NOTE: A & B & SUM refer to the MOST SIGNIFICANT DIGITS of the TWO NUMBERS BEING PROCESSED.

QUESTION:

4. Devise your own problem and then program the problem for solution on the computer.

5. Draw the logic required to provide an OVERFLOW LAMP on the computer.

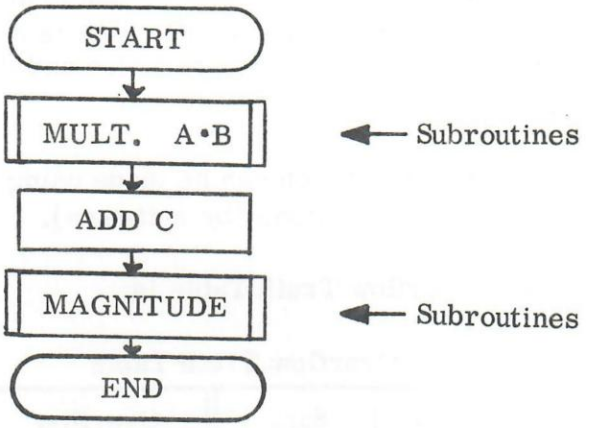
6. For the problem of X = A + B + C + D with only positive numbers; write the Symbolic Assembly Program for Overflow Detection.

DESCRIPTION (SUBROUTINES):

When a particular program routine (BCD to decimal, multiply, etc) is used many times, it is not necessary to repeat that program many times. The same routine is used over and over.

The flow chart is shown below using subroutines for the problem

$X = |A \cdot B + C|$



In order to use a "subroutine", the main routine must enter the "data" in specified address locations, and must enter an "exit" address in the subroutine.

This is illustrated below for a "magnitude" subroutine denoted as MAG.

"MAG" ROUTINE	= |A|	
L1	LA	LOAD
	L2	TR NEG
(L4)	L3	STORE
(L6)	__	TRANSFER
L2	0	LOAD
	LA	SUB
	L4	TRANS
LA	A	
L7	0	
L3	|A|	

The program starts at L1; determines MAG of the number located in LA; stores the result in L3; and continues with a next instruction at the address specified in the instruction in L6.

Consider the problem:

$X = |A| + B - 2|C|$

The program (using the subroutine) is given on the following page.

MAIN ROUTINE X

		Comments
L10	L17 LOAD	L1 thru L7 are used in the MAG routine.
	LA STORE	STORES IN SUBROUTINE
	L11 LOAD	L11 HAS [L12 TRANS] FOR EXIT FROM SUBROUTINE
	L6 STORE	STORE EXIT INSTRUCTION IN SUBROUTINE
	L1 TRA	TRANSFERS TO SUBROUTINE
	[MAG ROUTINE]	
L12	L3 LOAD	when subroutine is done, loads answer |A| in Accumulator
	L13 STORE	stores answer L13 → |A|
	L19 LOAD	repeats subroutine entry for C
	LA STORE	
	L16 LOAD	[L14 TRANS]
	L6 STORE	STORE EXIT TRANSFER INTO SUBROUTINE
	L1 TRA	EXIT FROM SUBROUTINE
	[MAG]	
L14	L13 LOAD	|A|
	B ADD	|A| + B
	L3 SUB	|A| + B - |C|
	L3 SUB	|A| + B - 2|C|
	L15 STORE	X
	HALT	
L11	L12 TRANS	exit constant
L16	L14 TRANS	exit constant
L17	A	
L18	B	
L19	C	
L15	X	
L13	|A|	

PROCEDURE (SUBROUTINES):

The above SYMBOLIC PROGRAM was prepared to obtain the solution for "X" and was prepared by an individual called a PROGRAMMER. The next step is to prepare the coded program. This can be prepared by another individual called a CODER or by a computer ASSEMBLY PROGRAM. The original equation may have been prepared by an individual known as the SYSTEM ANALYST who studied a particular problem.

Prepare the coded program for the above problem, enter it into the computer, and try some examples.

X = |A| + B - 2|C|				
TEST	A	B	C	X
1				
2				
3				

QUESTION:

7. Assume subroutines called MULT and DIV have been prepared. Write the MAIN ROUTINE to solve

$$Y = \frac{A \cdot B + 17}{C + 4}$$

DESCRIPTION (COMPUTER LANGUAGES):

The above programs are written using the computer instructions in the ASSEMBLY & MACHINE LANGUAGES. Easier programming can be accomplished by using "Higher Level" Languages such as FORTRAN, COBOL, BASIC, ALGOL, PL/I, etc. To solve the above problem provided in Question 7, a typical FORTRAN program is indicated below:

PROBLEM 7:

$$Y = \frac{A \cdot B + 17}{C + 4}$$

FORTRAN symbols are:

Plus	+
Minus	-
Multiplication	* (asterisk)
Division	/ (slash)
Exponential	**
Parenthesis	( ) indicates grouping as in mathematics
COSINE	(Indicates a special subroutine to solve for the cosine of an angle.)

The FORTRAN program statement for the solution of Y is:

FORTRAN PROGRAM STATEMENT

```
Y = ((A * B) + 17) / (C + 4)
```

Other FORTRAN statements include words such as:

READ (data from punched cards, tape, etc.)
PRINT
FORMAT (specifies output format for printing)
GO TO (transfers)
IF (conditional transfer)
STOP
DO (repeats program a number of times)
CALL (transfers to subroutines)
RETURN (transfers back to main program)

The program is entered on a full "typewriter" keyboard having all numbers, letters, and symbols. Each letter is coded with a binary code, and the program is stored in memory with one letter or number per 8 bit word (an 8 bit word is also called an 8 bit "byte").

A special computer program called a COMPILER, reads the FORTRAN binary codes from memory, and by examining the codes for the various functions *, +, -, **, /, determines the problem to be solved and compiles an assembly of standard routines to form the program routines in "machine language". The computer can compute only in coded machine language using instructions such as those used on the Ed - Lab computer.

QUESTIONS:

8. Write the FORTRAN STATEMENTS to solve the following equations:

a. $X = A + B^2$

b. $Y = (A + B)^2$

c. $Y = 3 \cos\left(\frac{A + B}{3}\right)$

d. $Z = \frac{-A \pm \sqrt{B^2 - 4AC}}{2A}$

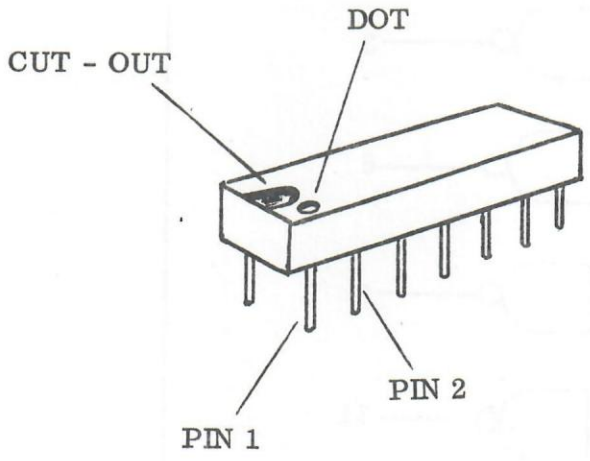
PRINTED CIRCUITS AND INTEGRATED CIRCUITS EXPERIMENT NO. 13

OBJECTIVE :

This experiment describes the use of the Printed Circuit (P.C.) socket and the Integrated Circuit (I.C.) socket to provide for further experimentation; to further the development of manual skills; and for experience in "logic troubleshooting and debugging ".

DESCRIPTION (I.C. SOCKET)

The Integrated Circuit Socket is for the "dual in - line" I.C.'s with 14 or 16 pins including both digital and analog modules. It can also be used for transistors or other elements with the leads connected to any convenient socket pins. The jacks on the panel are labeled 1 through 22 and are used for both the I.C. and P.C. sockets. The Integrated Circuit is inserted with the I.C. Pin Number 1 on the lower left hand side. Pin 1 on the I.C. is denoted by a small dot or cut - out. This is shown below:



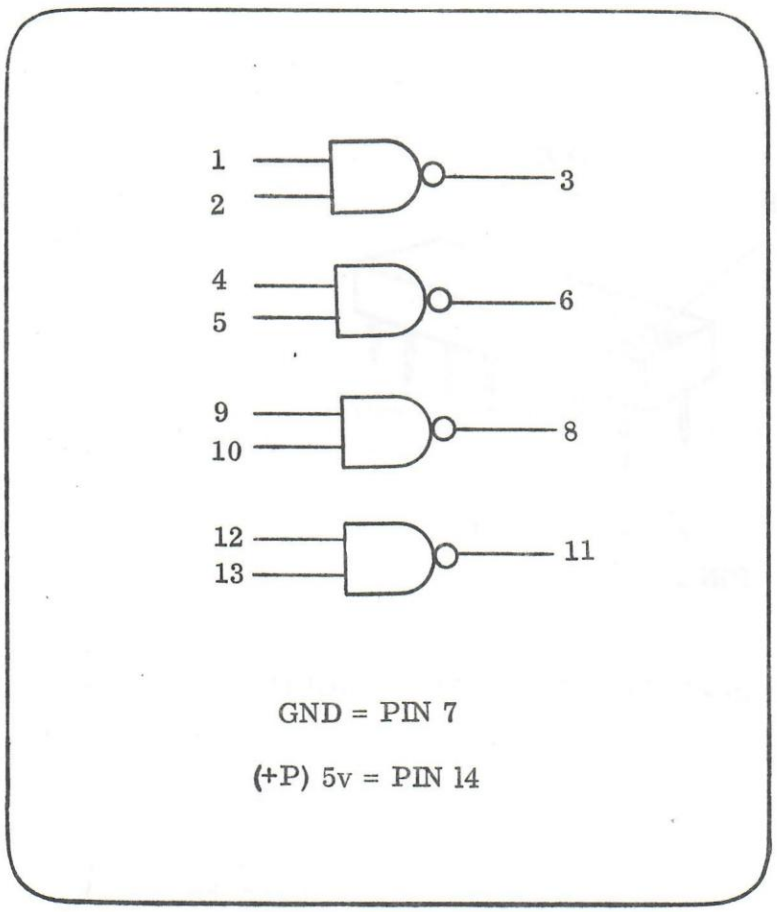
DUAL IN-LINE INTEGRATED CIRCUIT

A 16 pin I.C. is inserted into the socket and the jacks marked 1 through 16 correspond to the I.C. pins. The jack marked +P provides +5 volts via a 33 OHM series resistor; GND is ground; and -N provides -9 VOLTS via a 100 OHM series resistor. The series resistors provide a safety feature to prevent damaging any components by connecting power to a wrong pin. There will thus be a drop in the voltage applied to the module. If the module requires too much current, the voltage drop may be too large and the module may not operate. This should be no problem with normal I.C.s. The applied voltage can be checked when the module is connected to power.

A 14 pin I.C. may be placed in the socket to the extreme left. This leaves 2 pins open on the right (one bottom and one top). These two pins are normally labeled as 8 and 9. Thus, with a 14 pin module, the jack marked 10 is really I.C. pin 8 ; Jack 10 is thus marked 10 / 8 for convenience. Similar markings are used on 11 / 9 , 12 / 10 , etc., to 16 / 14 . The digit to the right hand of the "slash" is used with the 14 pin ICs.

PROCEDURE (I.C. MODULE) :

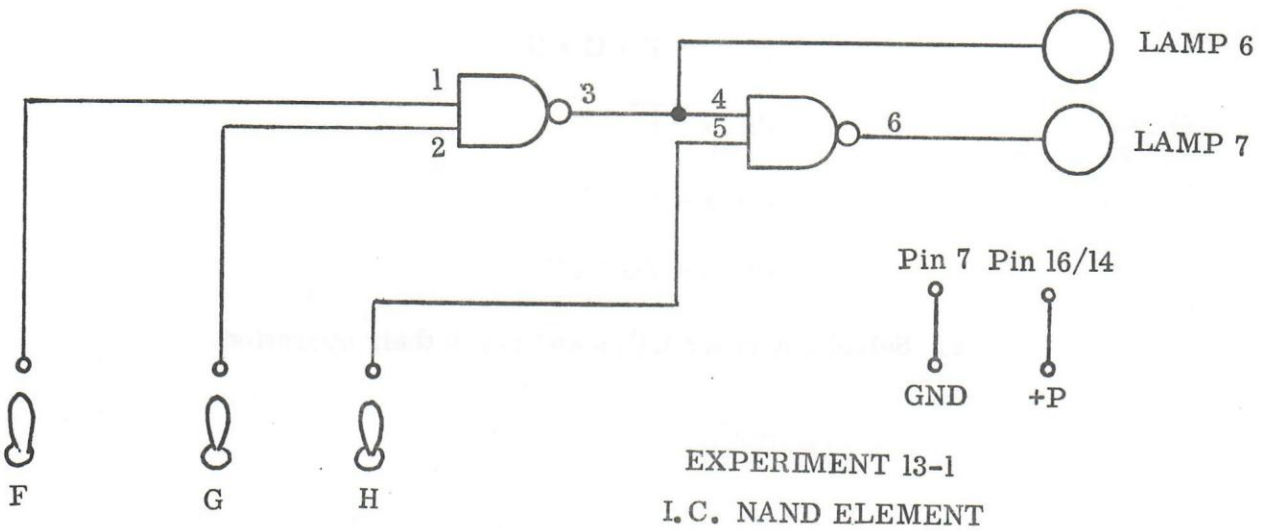
Insert a 7400 type I.C. module. The 7400 module is a "QUAD 2 - INPUT NAND GATE ". The module logic connections are shown below:



I.C. 7400

QUAD 2-INPUT NAND GATE

Connect the logic shown in the diagram below:



EXPERIMENT 13-1
I.C. NAND ELEMENT

Set the input switches and enter the outputs in the table below:

INPUT			OUTPUT		CHECK
F	G	H	6	7	
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

Check the outputs to verify proper operation.

EXERCISE :

1. Using the 7400, draw the logic for the following functions:

   (a) $x = F \cdot G \cdot H$

   (b) $x = \overline{F}$

   (c) $x = F + G$

   (d) $x = FG + JK$

2. Select any other I. C. s and check their operation.

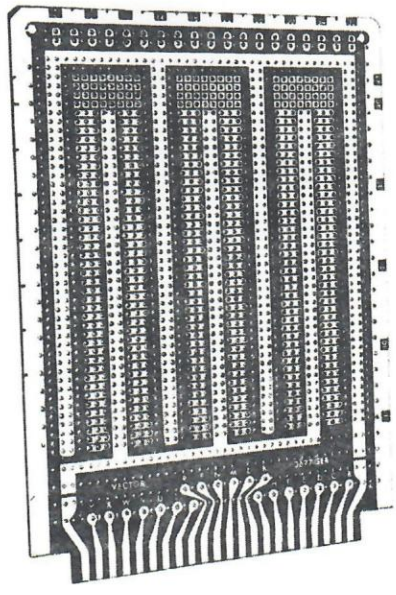
DESCRIPTION (P.C. CARD) :

The Printed Circuit Socket is a standard 22 pin, card edge socket with standard 0.156 inch centers on the contact "fingers". Standard cards are available from many sources including :

VECTOR ELECTRONIC CO., INC.
12460 Gladstone Ave.
Sylmar, California 91342

Phone (213) 365 - 9661

These are stocked by many electronic distributors. A typical "Vector board" is Part No. 3677 - 3. A typical integrated circuit socket, for mounting on the P.C. board, is Vector Part No. R714-1* for a 14 pin socket and Vector Part No. R716-1* for a 16 pin socket. Transistors, Capacitors, Resistors, etc., can also be mounted on the P.C. card. Students can thus build various projects to interface with the Ed - Lab equipment using the Ed - Lab internal power supply. It is recommended that an external power supply should NOT be used because incorrect wiring may burn-out components. The Ed - Lab power supplies have limiting resistors for full protection of the P.C. card and for the Ed - Lab trainers. A P.C. card is shown below:



TYPICAL PRINTED CIRCUIT BOARD

* (Package of 5 sockets.)

EXERCISE :

3. The students can obtain P.C. cards and modules, construct various logic combinations, and test these systems on the Ed - Lab 900. When I.C.'s are used in sockets on the P.C. card, faulty I.C.'s can be substituted and the students are required to find and repair the system in various logic troubleshooting and maintenance exercises.

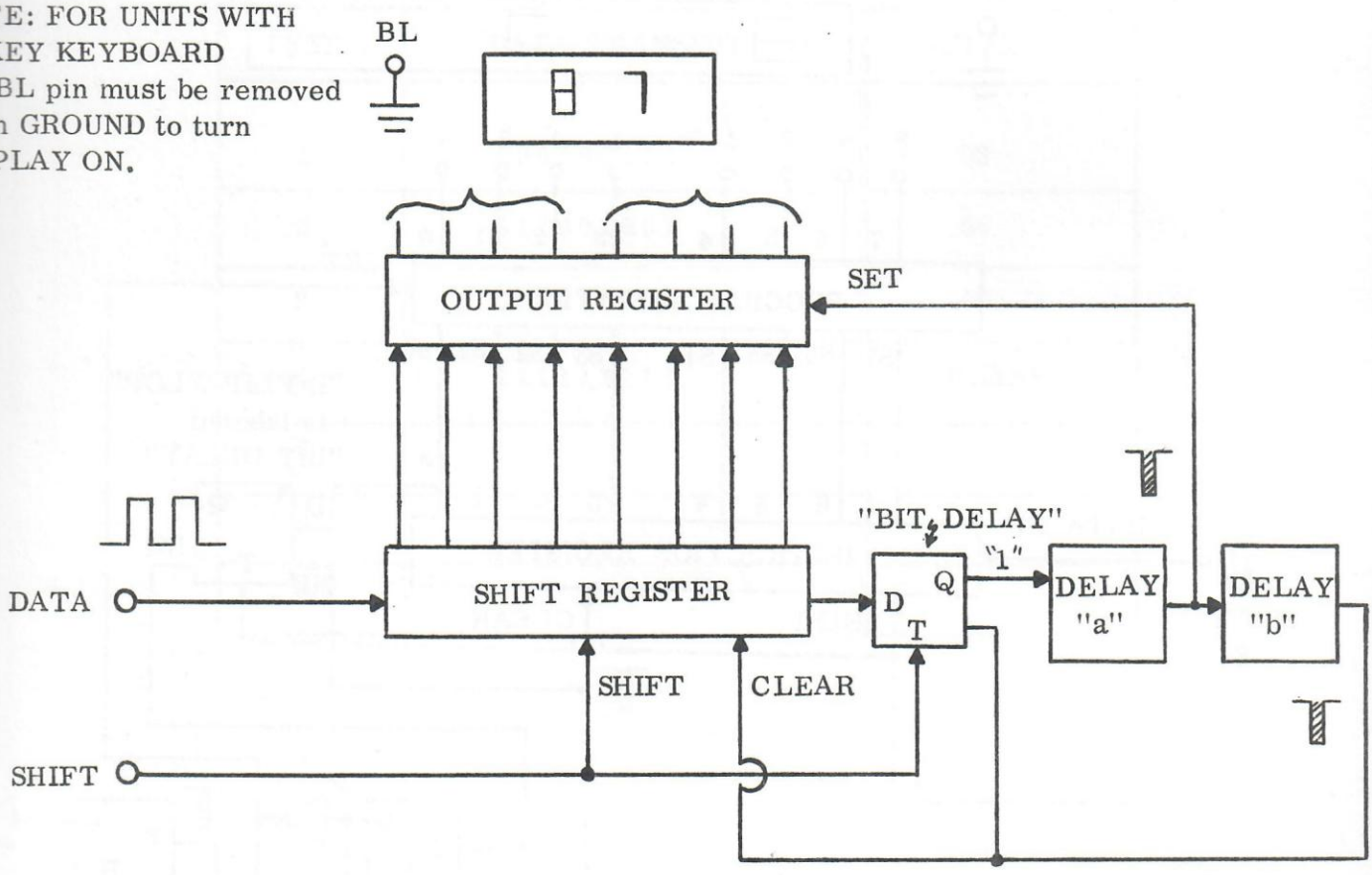
OBJECTIVE:

This experiment describes a serial to parallel data "refreshing" or "up-dating" system used in digital communications, displays and computer input-output and control systems.

DESCRIPTION:

Serial digital data is generally available from digital communications systems. The requirement for display and output is generally for a parallel presentation. Two data registers are required as shown below:

NOTE: FOR UNITS WITH 12 KEY KEYBOARD the BL pin must be removed from GROUND to turn DISPLAY ON.



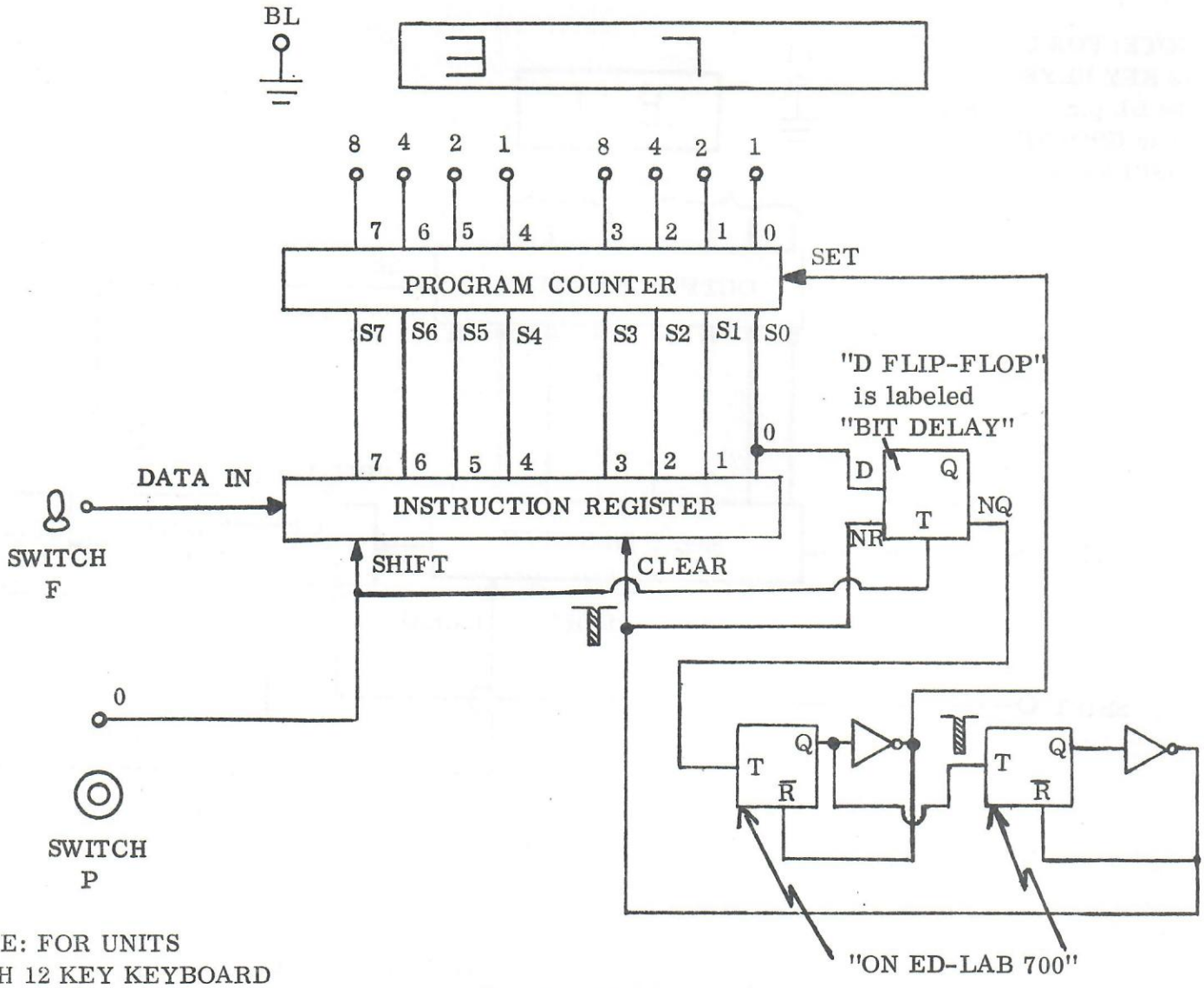
DATA UPDATING

The OUTPUT REGISTER holds the binary data to drive the display. The serial data is shifted into a SHIFT REGISTER. When all the data is entered, the new data is "transfered" to the REGISTER. In the system shown, an 8 bit data word is used. An extra ninth "1" bit is inserted at the beginning of the word. An extra bit (BIT DELAY on the Ed-Lab 900) is

inserted at the end of register. When the "1" is received in the extra BIT DELAY, it signifies that the full 8 bit word is in the SHIFT REGISTER. After a slight delay ("a"), a "transfer" or SET pulse transfers the data to the OUTPUT REGISTER. After another delay ("b"), a CLEAR pulse clears the SHIFT REGISTER.

PROCEDURE:

The Input Data is provided by Switch F, and the SHIFT synchronization is provided by momentary switch P. The logic is shown below:



NOTE: FOR UNITS WITH 12 KEY KEYBOARD the BL pin must be removed from GROUND to turn DISPLAY ON.

Experiment 14 - 1
DIGITAL OUTPUT REFRESHER

An Ed-Lab 900 and an Ed-Lab 700 are used. The two "delay" flip-flops are on the Ed-Lab 700. The PROGRAM COUNTER is used as the output register because only a SET input is required for data transfer. (A CLEAR and SET would be required on the ARITHMETIC REGISTER.) The two flip-flops on an Ed-Lab 700 are connected to generate short pulses for SET and CLEAR.

Wire the logic. Serially enter data by using Switch F to set data and Switch P to shift the data. Transmit 9 bits with a "1" as the first bit. Check the output display and verify "refreshed" data. Enter results in the table below:

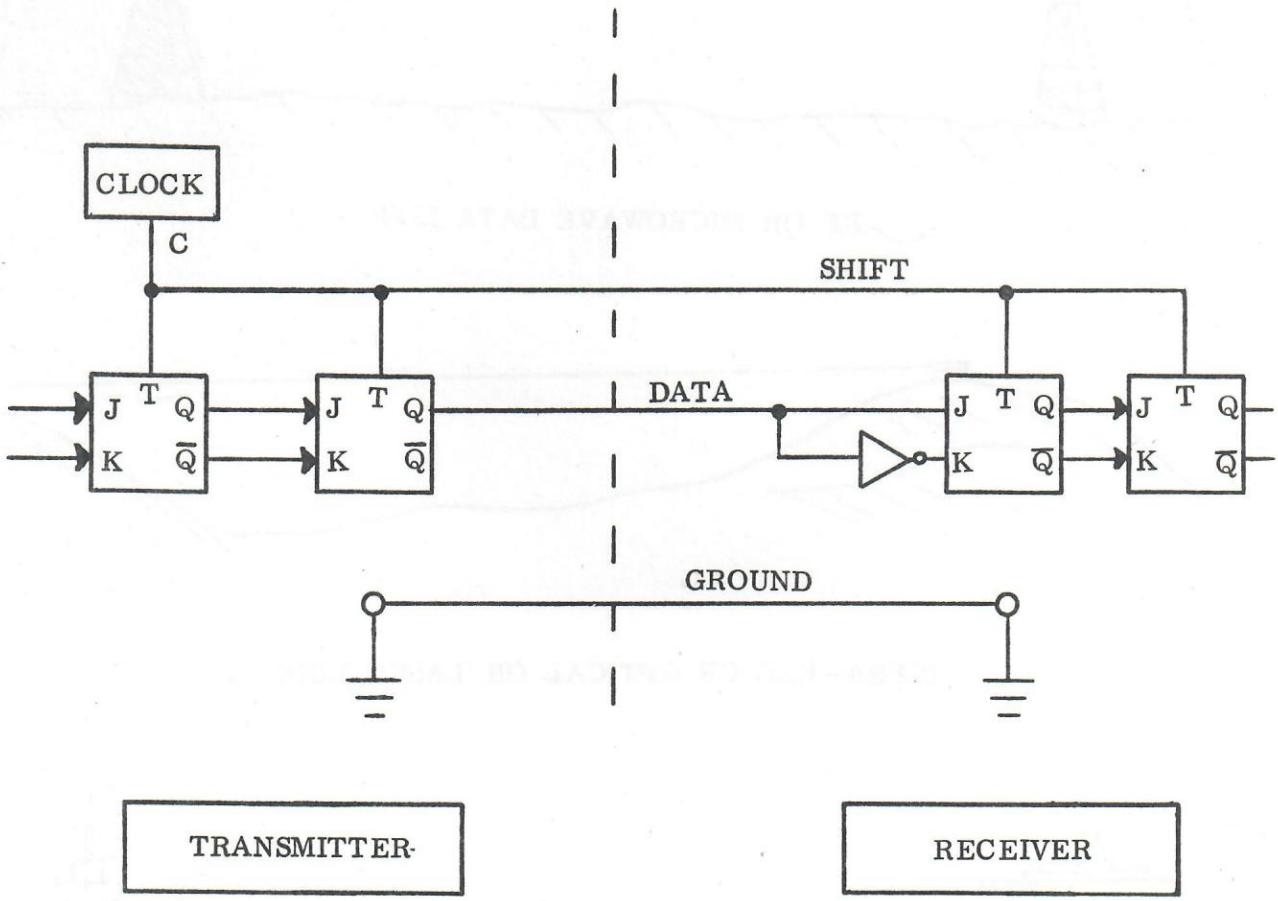
TEST	DATA TRANSMITTED	OUTPUT
1	100100111 ← FIRST	93
2	011001001	64
3	000000001	00
4	111111111	(BLANK)
5		
6		
7		

OBJECTIVE:

This experiment demonstrates how the communication of data is performed using a "digital data link". Various techniques for coding of digital 1's and 0's are presented and a complete transmitter and receiver MODEM link is established.

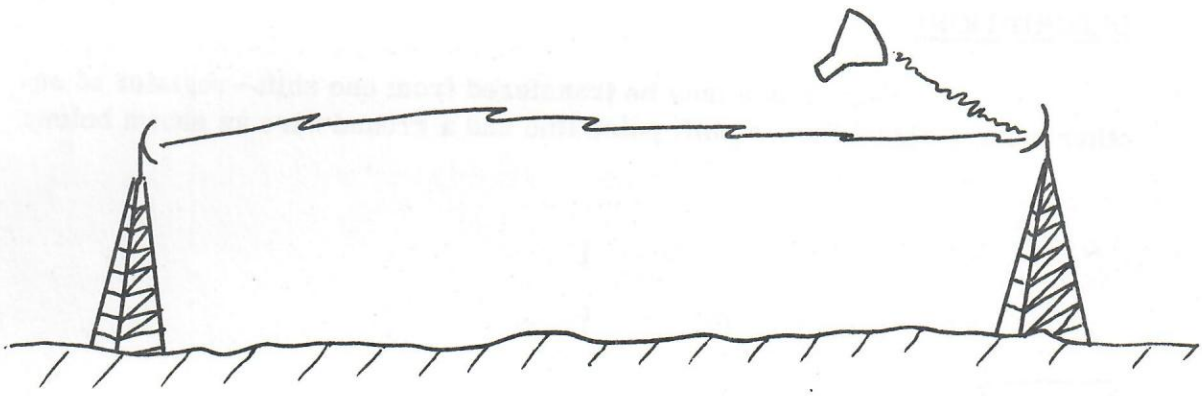
DESCRIPTION:

Digital data may be transfered from one shift - register to another using a signal line, a shift pulse line and a ground wire as shown below:

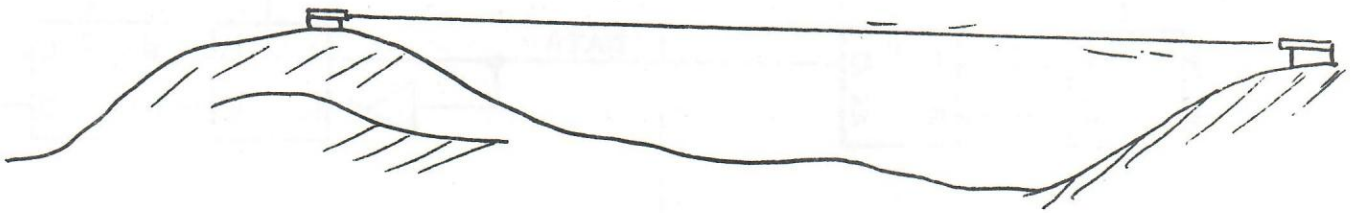


DATA AND SHIFT LINES

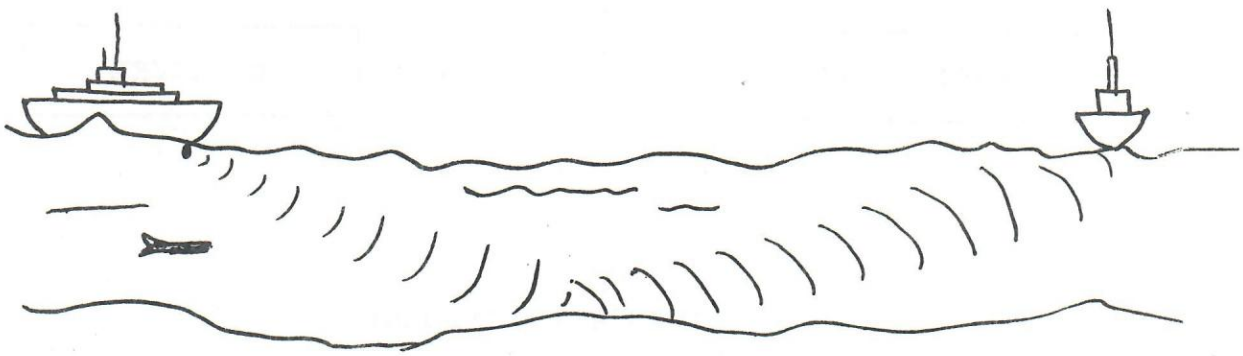
However in transmitting over long lines, delays will occur between the SHIFT signal and DATA, and generally only one transmission line or channel is available. The proper shift trigger must then be generated at the receiver. Typical Data Links are:



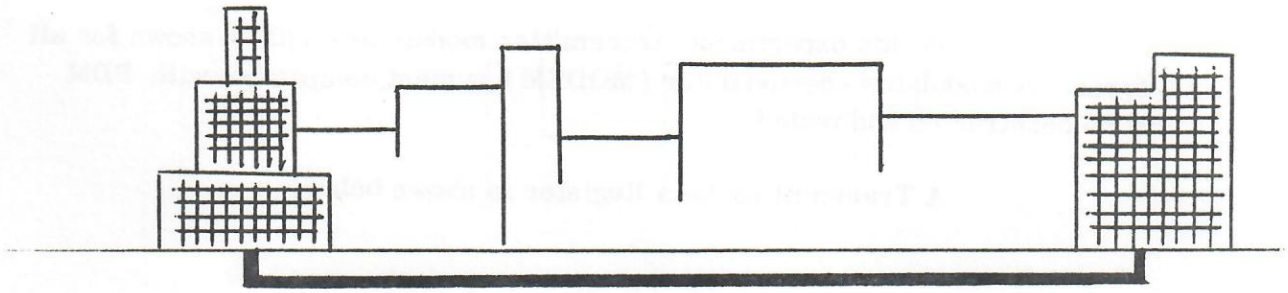
RF OR MICROWAVE DATA LINK



INFRA-RED OR OPTICAL OR LASER LINK

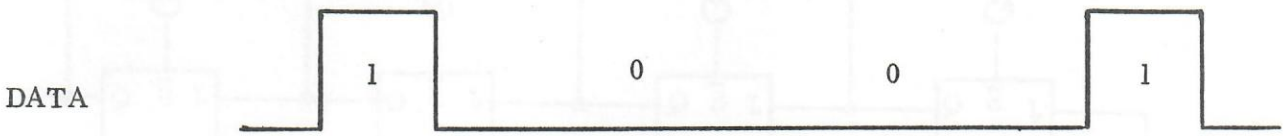


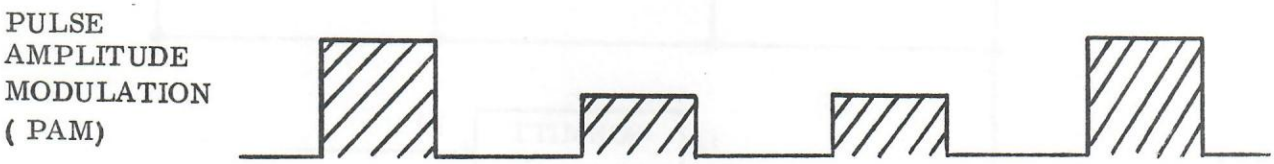
ACCOUSTIC OR AUDIO (ULTRA SONIC) LINK

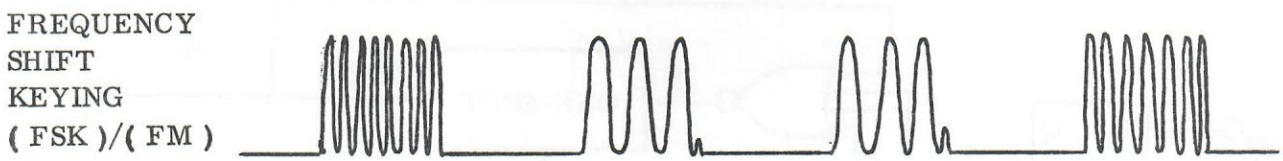


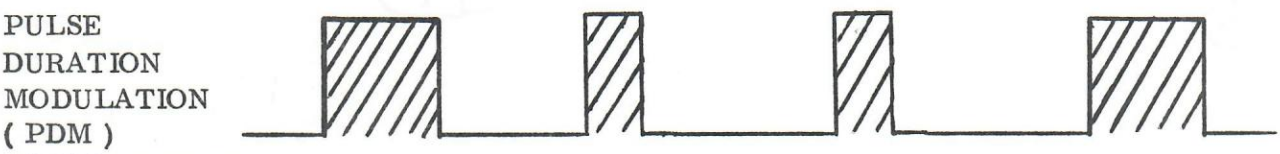
CABLE LINK

The binary 1's and 0's can be coded in various ways, typically:





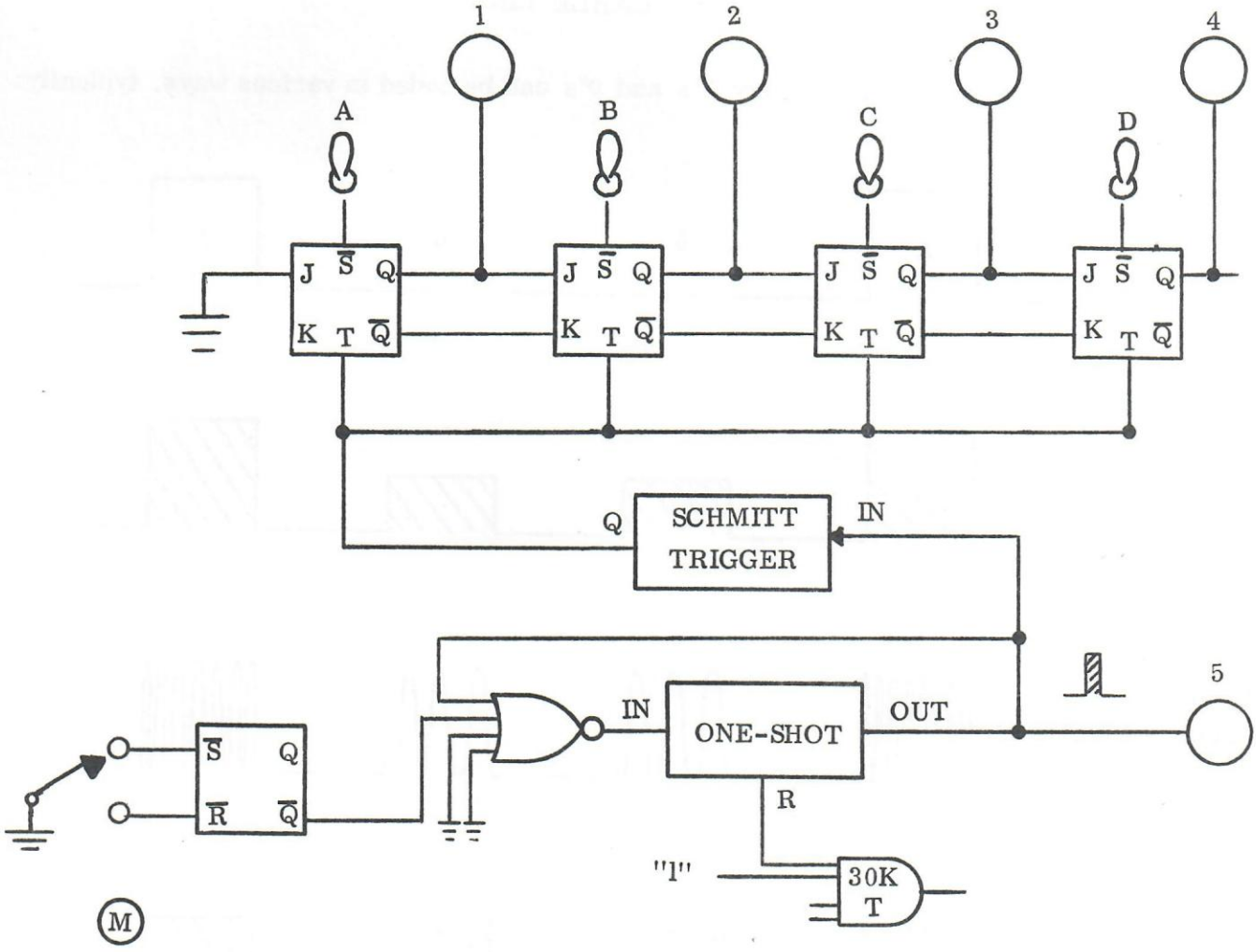




PULSE CODE MODULATION

In this experiment, transmitter modulators will be shown for all 3 types. A modulator - demodulator (MODEM) system compatable with PDM will be constructed and tested.

A Transmitting Data Register is shown below:



EXPERIMENT NO. 15-1

TRANSMIT DATA REGISTER

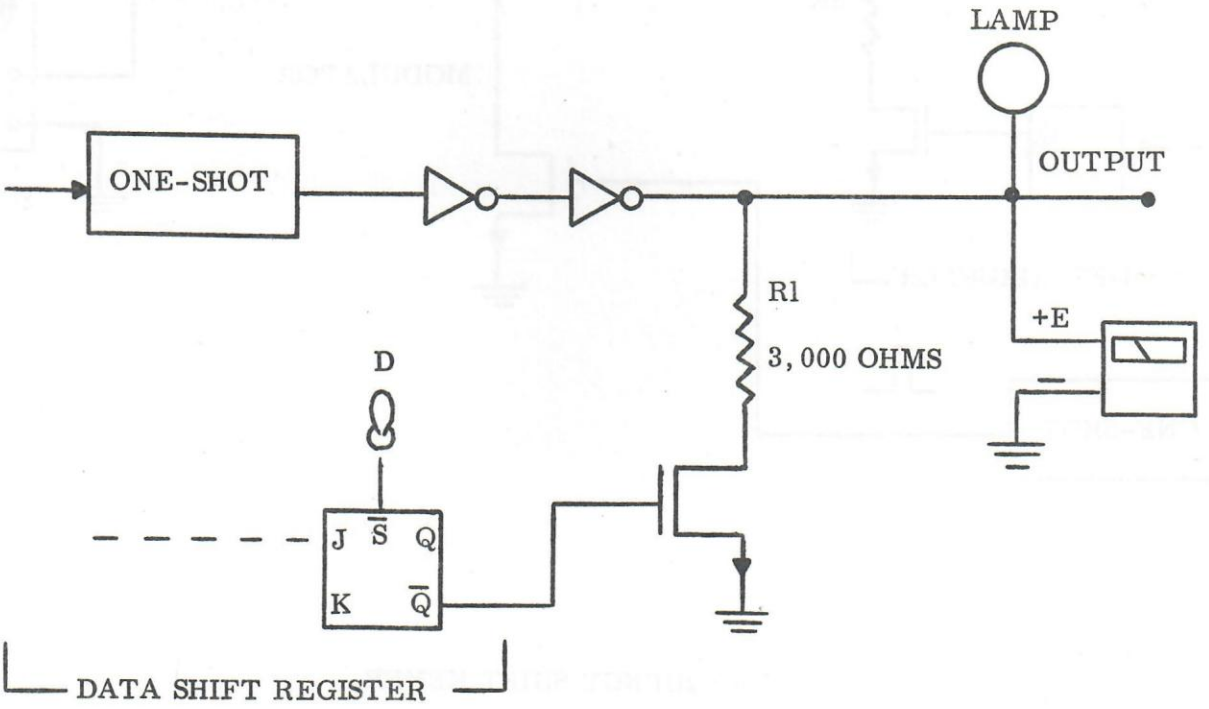
When Switch M is depressed, the One-Shot Pulse is generated. When the pulse goes to "0", the register is shifted. The input flip-flop J is connected to "0" for input clearing while shifting. Data is set by using Switches A, B, C, D in a momentary Down-then-Up operation. The NOR gate is used to enable the wide One-Shot pulse to operate. If the NOR were not used, the output pulse would be shut-off when the input pulse was cut-off.

PROCEDURE:

Wire the logic shown in Experiment 15 - 1. Depress M 4 times to clear the register. Set in any number using Switches A, B, C, and D. Pulse Switch M 4 times and observe the generation of a "One-Shot" Pulse on Lamp 5.

DESCRIPTION (Pulse Amplitude Modulator):

The pulse amplitude can be decreased by "loading" a buffer amplifier. Thus using 2 inverters and a solid state switch, the circuit is:

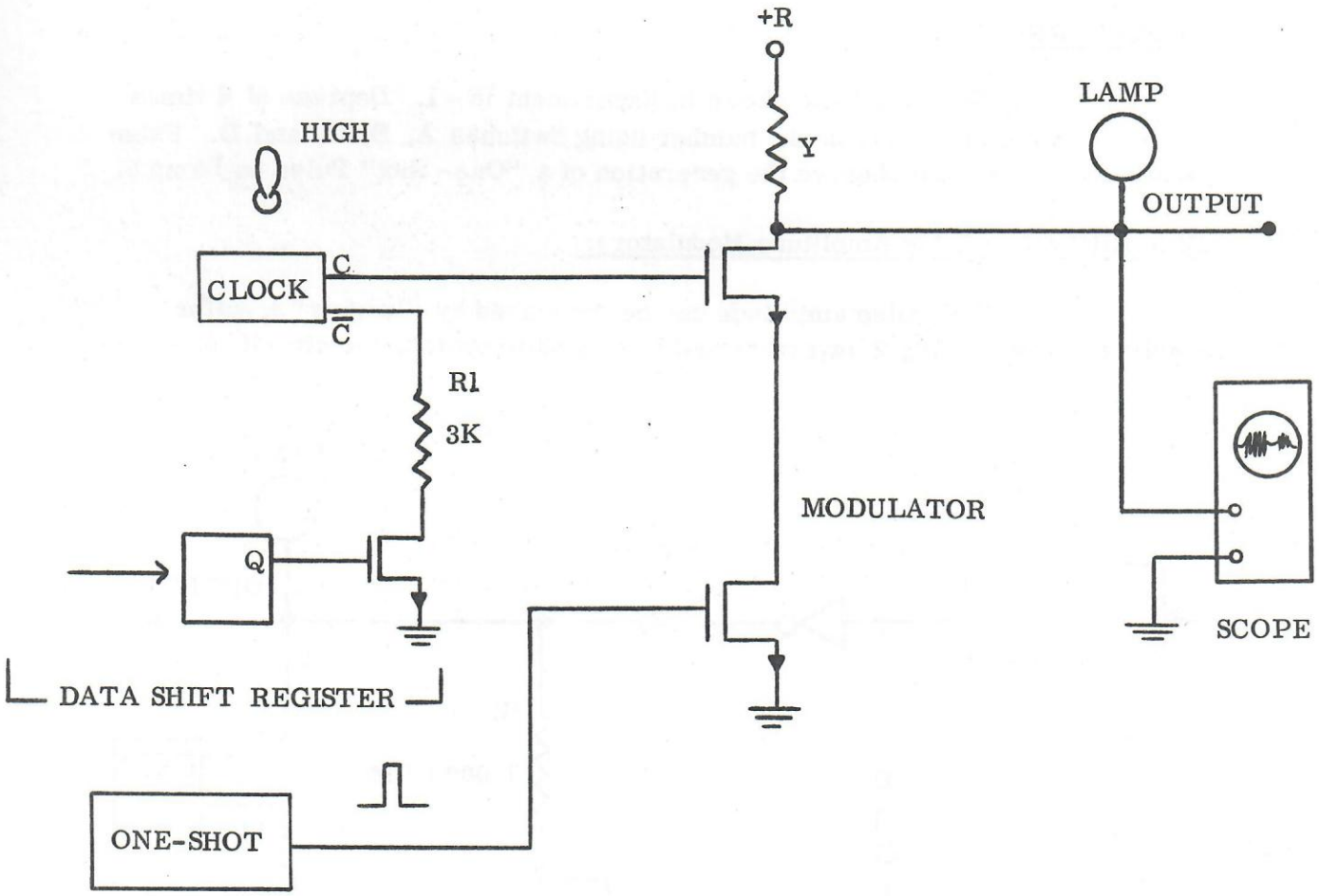


Logic as in Exp. No. 15-1

PULSE AMPLITUDE MODULATOR

DESCRIPTION (FSK Modulator):

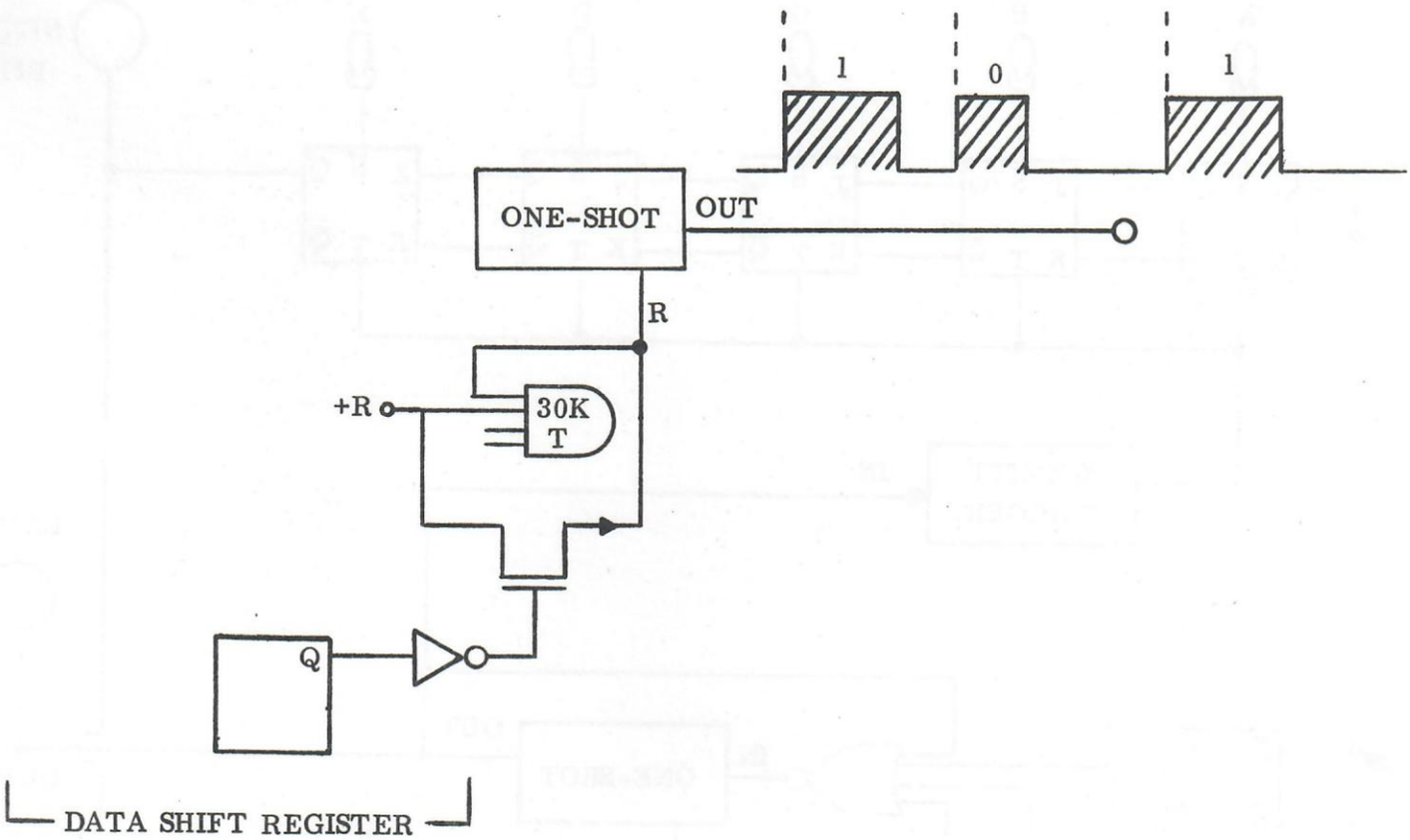
Two solid-state switches are used to "modulate" a higher frequency clock. This system is shown below:



FREQUENCY SHIFT KEYER

DESCRIPTION (PDM MODULATOR):

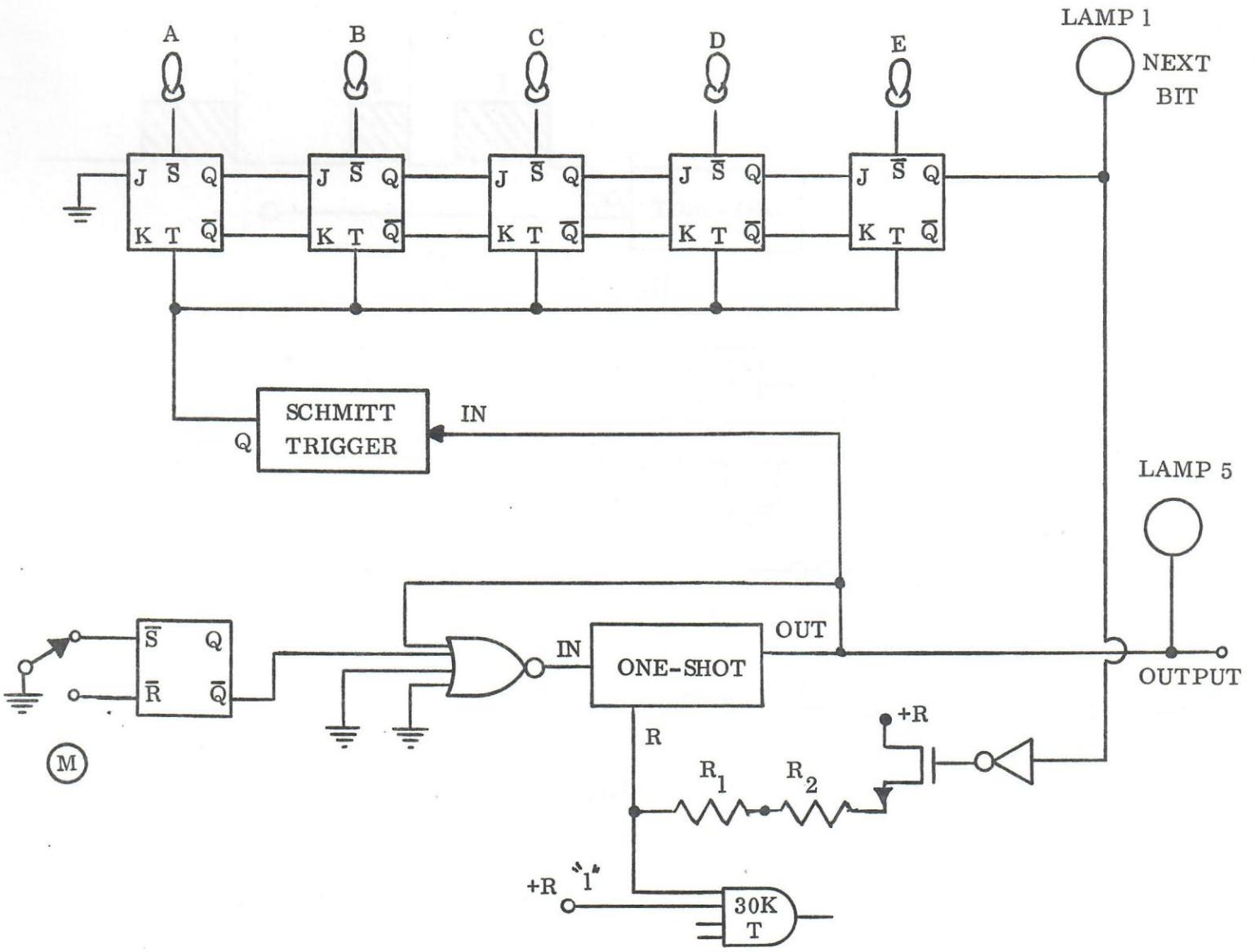
The variation in pulse width is controlled by varying the One-Shot resistor as shown below:



PDM MODULATOR

The output data register controls a solid-state switch to "short" the external 30,000 ohm resistor and to thus create a shorter pulse on a "0" data signal.

The complete logic for the PDM Modulator - Transmitter is given below:



EXPERIMENT 15-2

PDM MODULATOR- TRANSMITTER

PROCEDURE (PDM Transmitter):

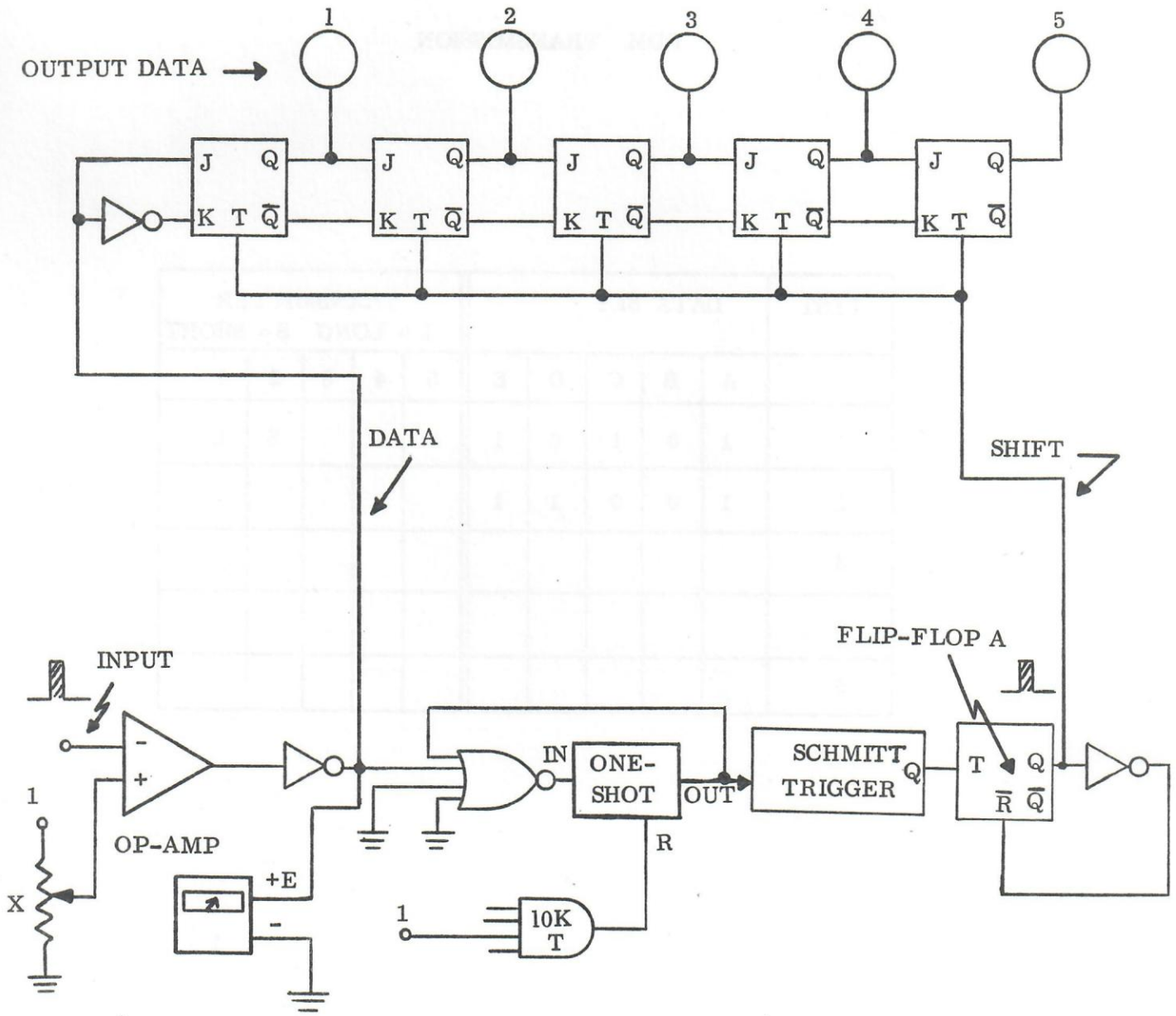
Wire the PDM system shown in Figure 15-2. Shift 5 times using M to clear the data register. Use the 5 switches A, B, C, D, E to set in data (down-then-up). Use M to shift out data and observe the output pulse widths using Lamp 5. Enter the data below:

PDM TRANSMISSION

TEST	DATA SET					TRANSMITTER L = LONG S = SHORT				
	A	B	C	D	E	5	4	3	2	1
1	1	0	1	0	1				S	L
2	1	0	0	1	1					
3										
4										
5										

DESCRIPTION (PDM DECODER):

The Receiver or Decoder for a PDM system must determine a "long" or a "short" pulse. The logic is shown below:



NOTE: 1 indicates connection to TI BUS (reference Voltage)

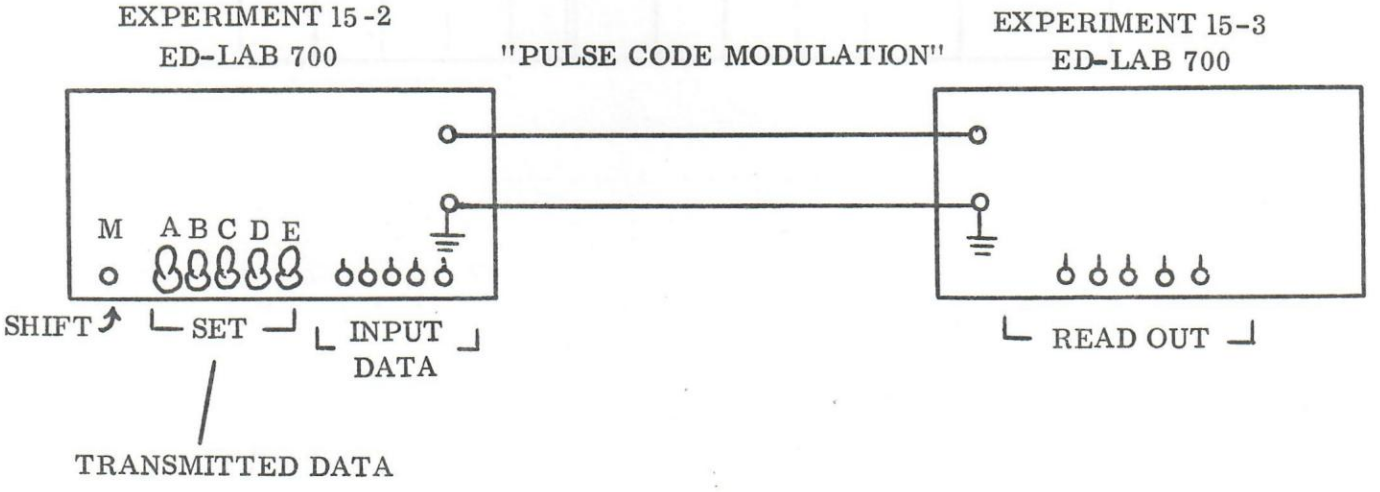
EXPERIMENT 15-3

PDM DECODER

The Input Signal (Long or Short Pulse) is applied to the Operational Amplifier. The X Voltage Level sets a Threshold to eliminate any "offset" or "noise" on the incoming signal. The One-Shot is adjusted to a medium width (between the long and short binary "0" and "1" level using the 10K resistor). The NOR permits a pulse of any width and the Schmitt Trigger generates a "sharp" pulse which triggers Flip-Flop A generating a "1" at Q. The inverter produces a reset pulse which resets Q to 0. The Q output is thus a short pulse, appearing at the end of the one-shot pulse, which shifts the shift register. The shift register input flip-flop is "steered" (J, K) by the input data pulse. Thus, when the "shift" pulse occurs, a "wide" input pulse provides a "1" at "J" and a "narrow" pulse provides a "0" at "J", and the proper "bit" (1 or 0) is shifted into the register.

PROCEDURE (PDM MODEM):

Use one Ed-Lab 700 for the PCM Modulator and one Ed-Lab 700 for the Demodulator (MODEM = Modulator-Demodulator). Separate the two units with 2 long wires as shown below. Wire the two units as shown below and in Experiment 15-2 and 15-3:



PDM MODEM

While observing the meter, adjust POT X until the meter reads 0, plus an extra threshold tolerance on POT X.

Shift M five times to clear the input register. Enter transmit data using the five Switches A, B, C, D, E. Select some codes and transmit the data using Switch M. Enter the data below:

TEST	TRANSMIT					RECEIVE				
	A	B	C	D	E	1	2	3	4	5
1										
2										
3										
4										
5										

QUESTIONS:

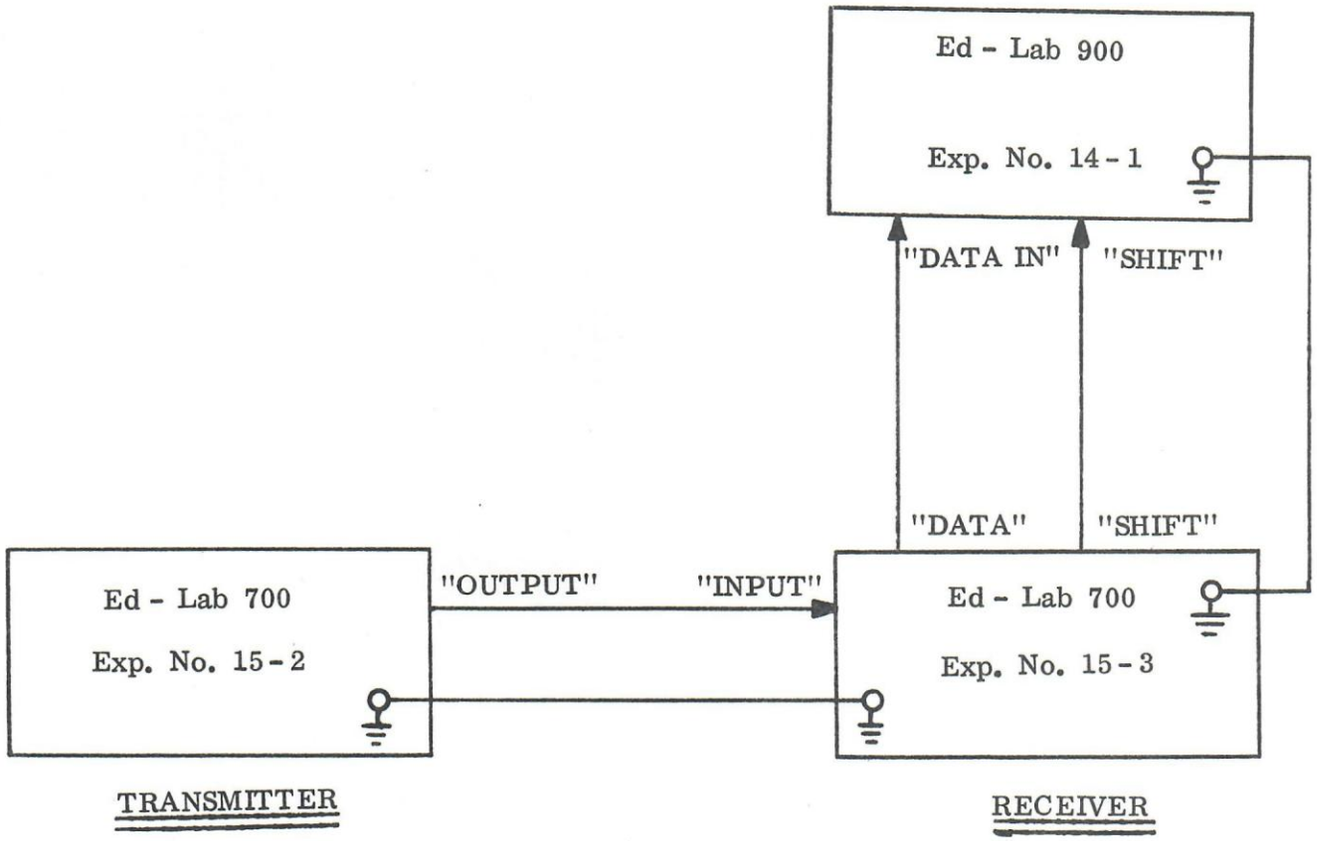
1. Draw the logic for a PAM (Pulse Amplitude Modulation) Demodulator. (Use the op-amp as a threshold comparator.)

2. As an exercise, the student can operate a PAM Modem.

DESCRIPTION (Up - date Communications System):

An Ed - Lab 900 can be added to provide a remote data communications system with continuous display updating.

The system is shown below and combines this experiment with the previous experiment.



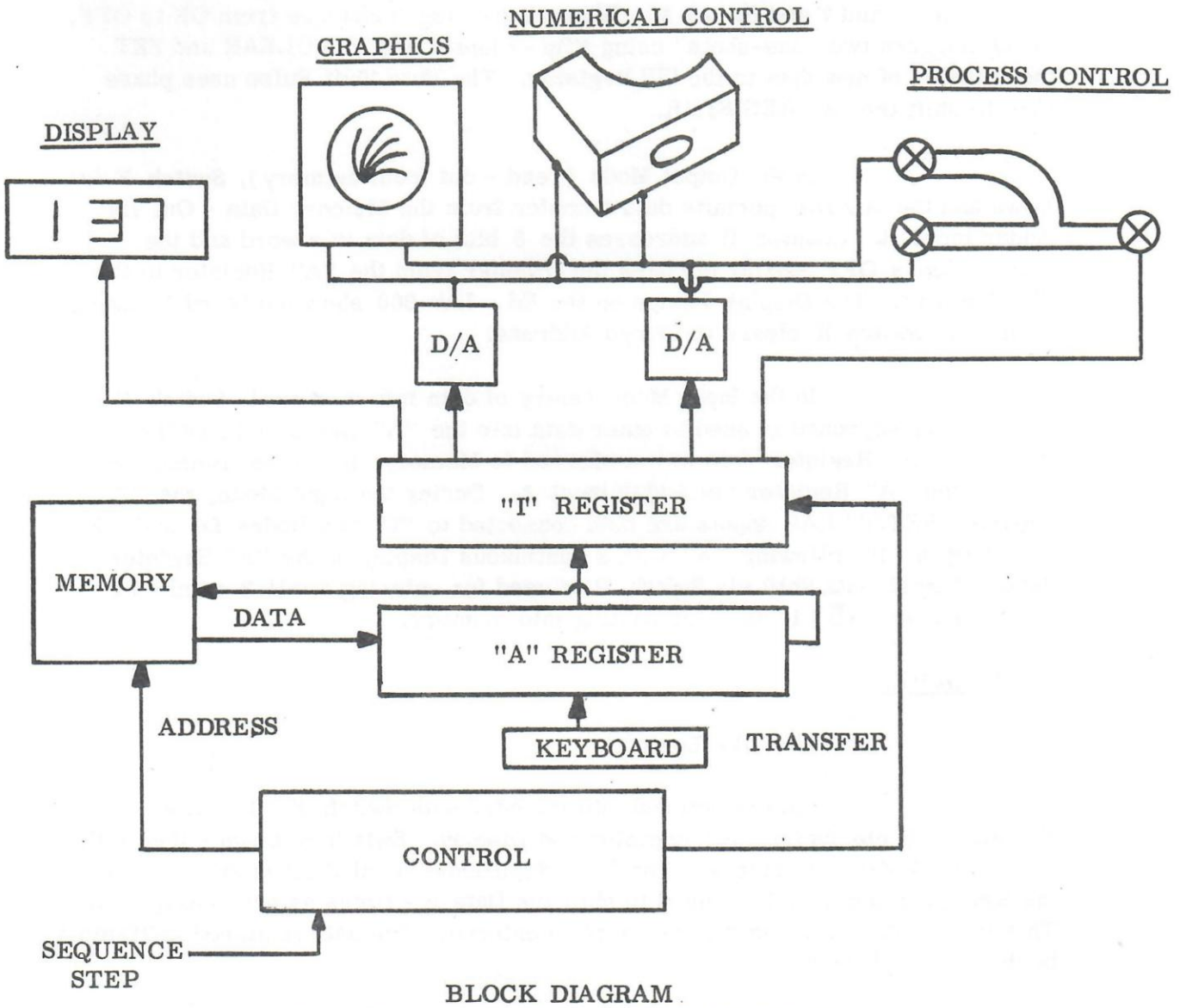
Experiment 15 - 4
REMOTE DISPLAY SYSTEM

The student can wire the entire system for remote displays. The 5 flip - flop shift register in Diagram 15 - 3 is not required and is replaced by the Instruction Register in Diagram 14 - 1.

OBJECTIVE:

The 256 bit MEMORY can be used to store a preset series of numbers which can then be provided in sequence as a parallel output to control various systems. Typical systems include DISPLAYS, NUMERICAL CONTROL machine tools, COMPUTER CONTROLLED PROCESSES, Control of AUDIO SYSTEMS, and so on. This experiment provides a means for storing data and providing a controlled sequential output for a NUMERICAL CONTROL DISPLAY.

Typical applications for a sequential memory are indicated in the Block Diagram below where a continuous voltage is obtained from a computer memory, and controls a process.



BLOCK DIAGRAM
SEQUENTIAL MEMORY BLOCK DIAGRAM

The data is transferred serially from MEMORY to an Intermediate Register, "A", and then the new word is transferred in parallel to the Output Register, "I". This output can then be used to drive a Display, or via Digital to Analog Converters (D/A) to drive Graphic Displays, Motors, Control Valves, etc. The Keyboard enters data into the "A" Register for "loading" of the Memory.

The detailed logic diagram is shown on the following page. The Ed - Lab 700 is used in conjunction with the Ed - Lab 900.

The Clock triggers Flip-Flops 1 and 2 to provide the four phases for recording, shifting, and counting as in the Computer Control Unit. The Bit Counting is performed using the 3-bit Counter B whose output triggers the OFF Flip-Flop 9 and the Word Address Counter P. Starting is via Flip-Flops 6 and 7 and Switch M. When Flip-Flop 9 changes from ON to OFF, the Q triggers two "one-shots" using Flip-Flops 4 and 5 to CLEAR and SET the transfer of new data to the "I" Register. The Data Shift Pulse uses phase (AB) to shift the "A" REGISTER.

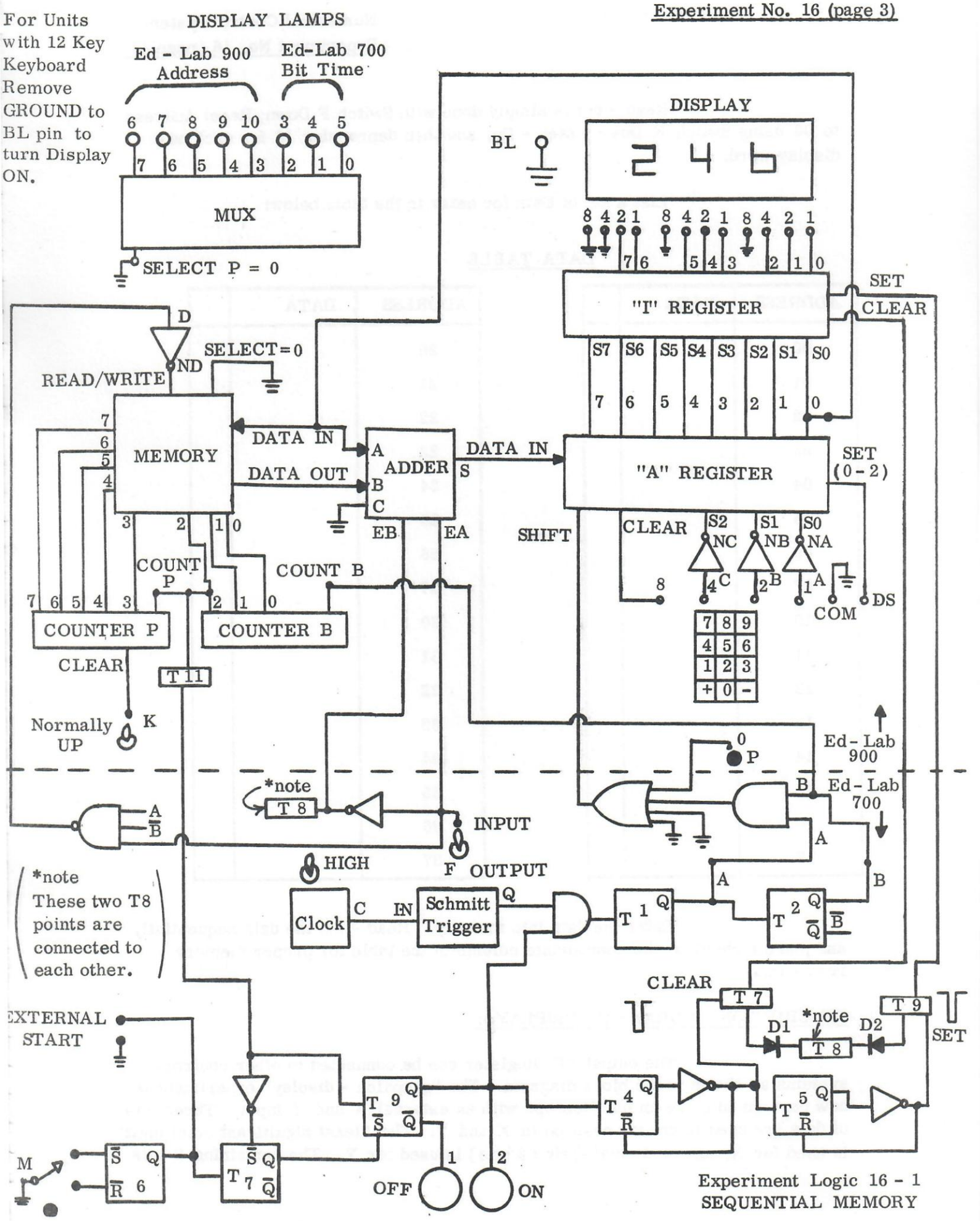
In the Output Mode (read - out from memory), Switch F is Down and the ADDER permits data transfer from the Memory Data - Out via Adder input B. Counter B addresses the 8 bits of data in a word and the Flip - Flop 9 OFF setting initiates the transfer from the "A" Register to the "I" Register. The Display Lamps on the Ed - Lab 900 show the Word Location Address. Switch K clears the Word Address.

In the Input Mode (entry of data into memory), Switch F is Up. The Keyboard is used to enter data into the "A" Register in OCTAL. When the "A" Register data is transferred to Memory, it is also transferred back to the "A" Register via Adder input A. During the Input Mode, the "I" Register SET/CLEAR inputs are both connected to "0" via diodes D1 and D2 resulting in "I" following "A" with a continuous Display of the "A" Register data. Manual Data Shift via Switch P is used for entering a full 3 octal-digit word. Phase ($A\bar{B}$) is used for writing into memory.

PROCEDURE:

Wire the Logic.

Data is entered into memory with Switch F UP. The Keyboard #8 clears the "A" Register and Display. Switch K Down - then - Up, clears the Address Counters. The Most Significant Octal Digit is entered via the Keyboard and then P is used to shift the Data 5 - times as previously used. This is repeated until the desired word is entered. The data is stored in Memory by depressing Switch M.



Experiment Logic 16 - 1
SEQUENTIAL MEMORY

Read - out is simply done with Switch F Down; Reset Address to 00 using Switch K Down - then - Up, and then depressing M for each next display word.

List a set of Data for entry in the table below:

DATA TABLE

ADDRESS	DATA	
00		
01		
02		
03		
04		
05		
06		
07		
10		
11		
12		
13		
14		
15		
16		
17		

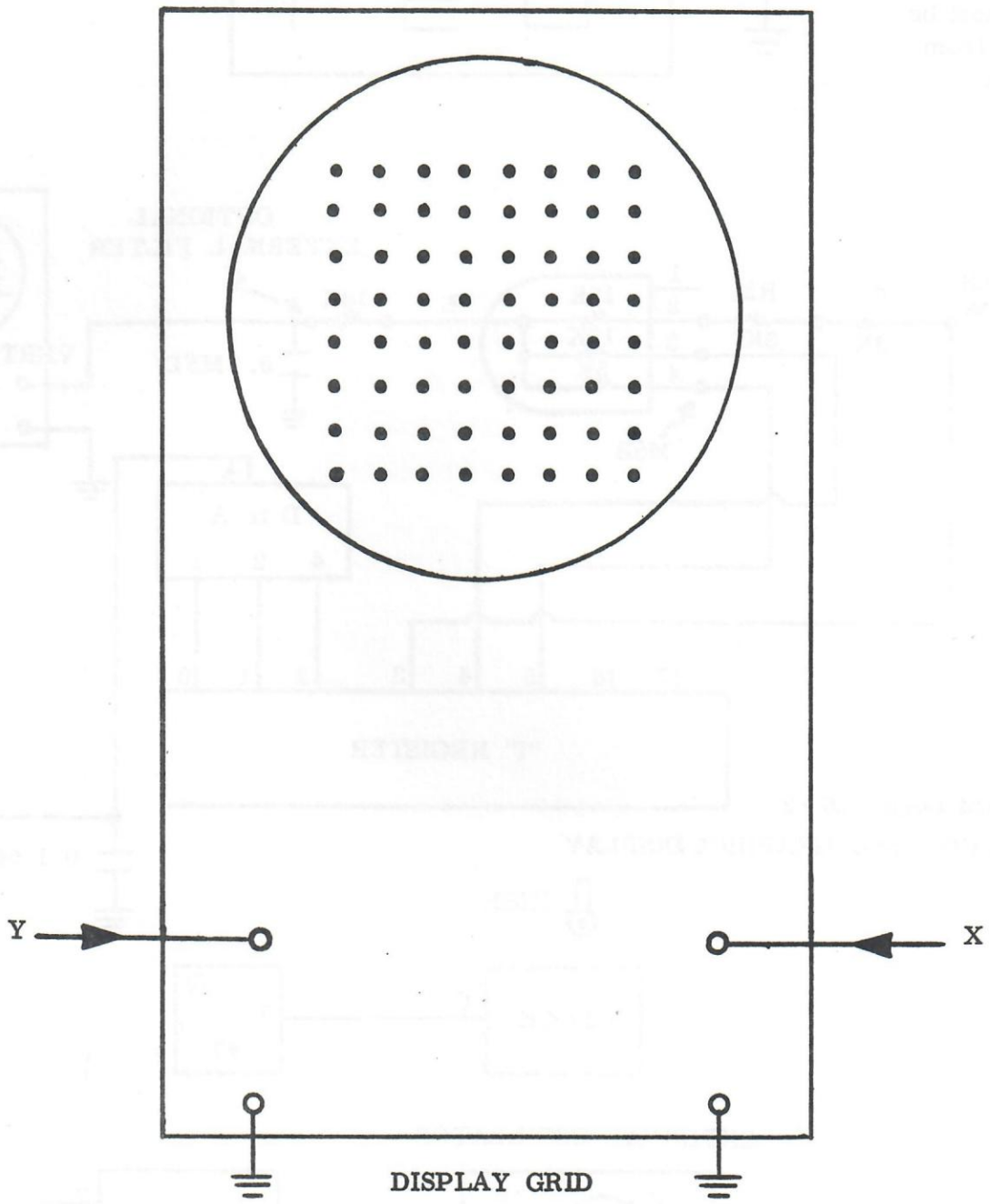
ADDRESS	DATA	
20		
21		
22		
23		
24		
25		
26		
27		
30		
31		
32		
33		
34		
35		
36		
37		

Enter the data into memory. Read - out the data sequentially and place a check in the appropriate column of the table for proper memory read - out.

DESCRIPTION (GRAPHICS DISPLAY):

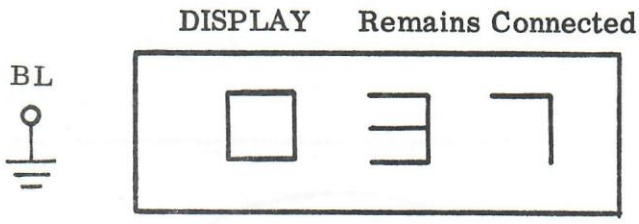
The output "I" Register can be connected to other control systems as shown in the block diagram. The "graphics - display" experiment is now performed using an oscilloscope with an external X and Y input. Three bits of data are used to decode position in X and Y. The "least significant octal digit" is used for X, and next octal digit (3 bits) is used for Y. The remaining 2 bits

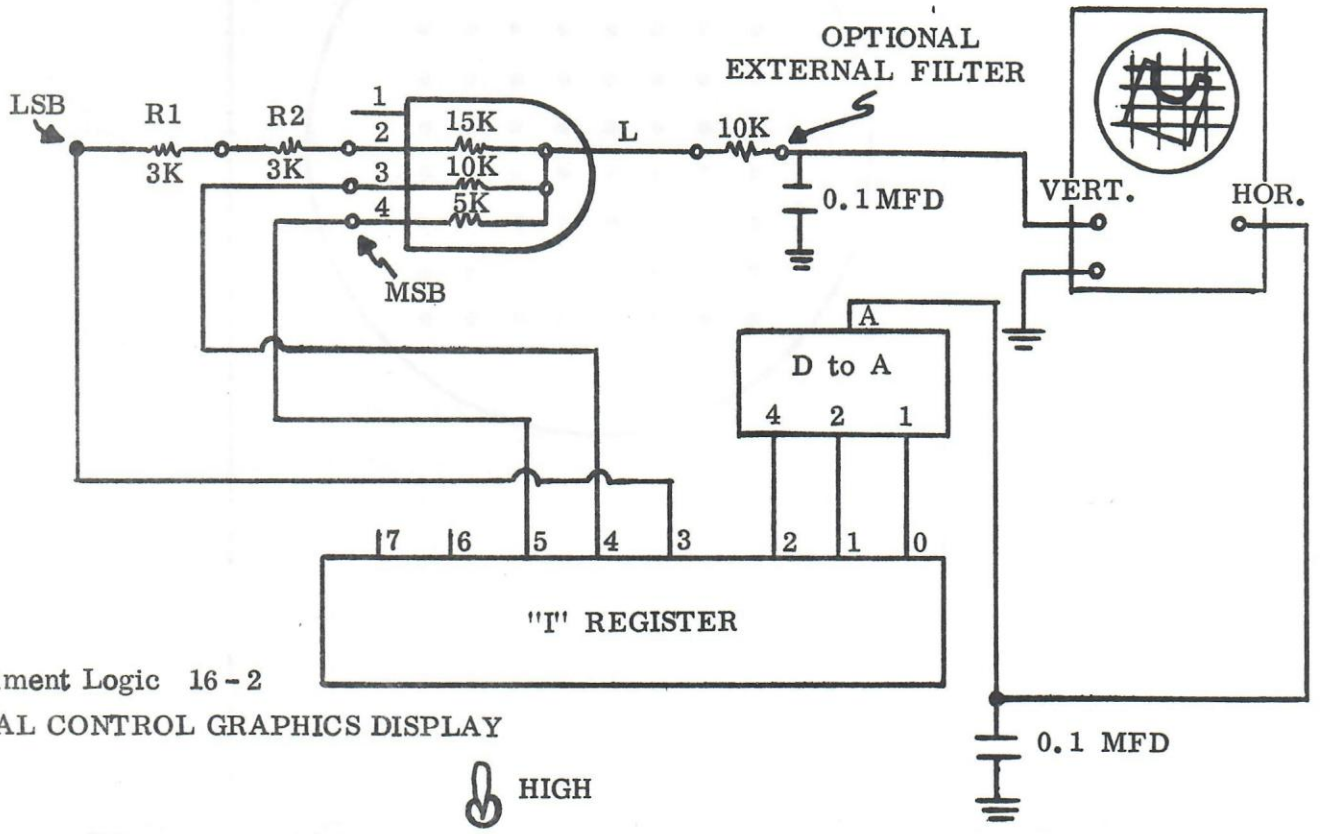
can be used for control, blanking, etc., but will not be used at this time. The Scope presentation is thus divided in an 8 by 8 grid of 64 points as shown below:



The analog converter output logic is shown below:

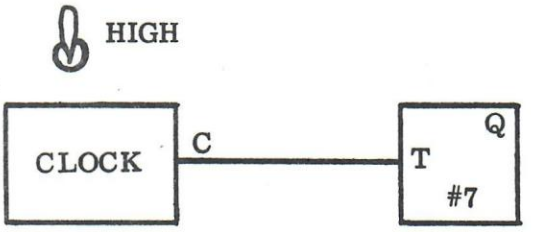
NOTE: FOR UNITS WITH 12 KEY KEYBOARD the BL pin must be removed from GROUND.

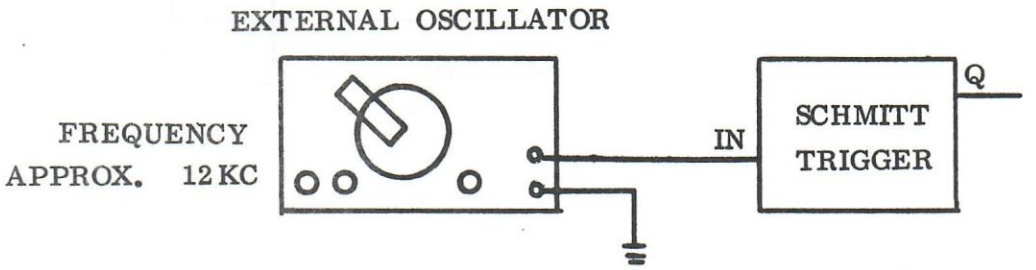




Experiment Logic 16 - 2

NUMERICAL CONTROL GRAPHICS DISPLAY



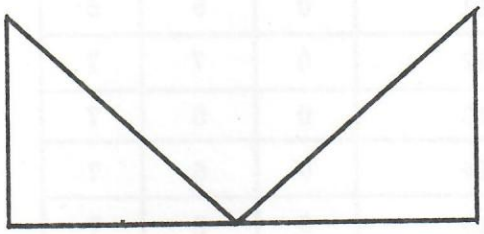


The D to A Converter is used for the horizontal "X" drive. The Threshold Logic resistors are used for the "Y" drive digital to analog conversion. The Display remains connected. The analog outputs and ground are connected to an oscilloscope. The Ed - Lab 700 CLOCK is disconnected from the logic shown in 16 - 1 and is connected to Flip - Flop #7 Trigger, T, to automatically provide a rapid sequence. An external oscillator (clock) is used for a fast read - out of memory data to yield a rapidly repeating "picture" for a "continuous" presentation on the 'scope. Temporarily disconnect the clock from Flip-Flop 7.

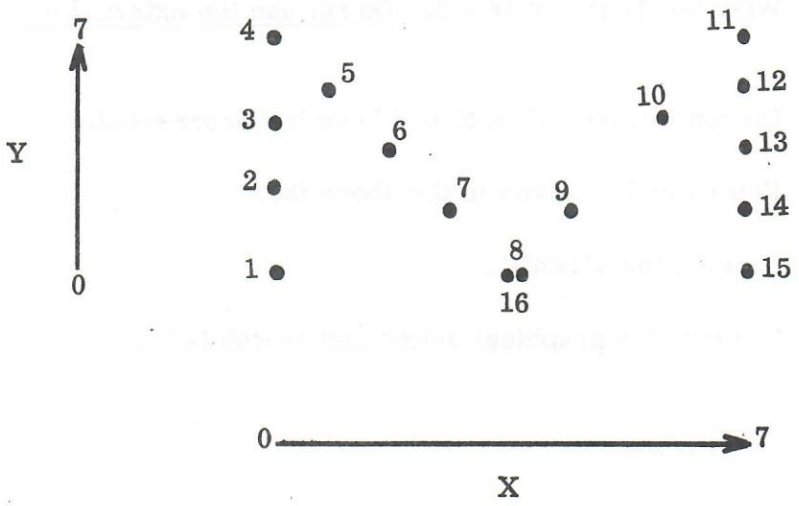
The data is entered into MEMORY in the normal manner (as in Figure 16 - 1). Reconnect the clock to the Trigger of Flip-Flop 7. Fast read-out will then result in a graphical display on the oscilloscope. The external capacitors and 10 K resistors are used for "smoothing" between points.

PROCEDURE:

A graphical example is given below:



This shape will be reproduced on the "screen" by selecting 16 points on an 8 x 8 grid as shown below:



The memory will use half of its capacity by storing the 16 points as shown below:

ADDRESS	DISPLAY		
		Y	X
00	0	0	0
01	0	2	0
02	0	5	0
03	0	7	0
04	0	5	1
05	0	3	2
06	0	2	3
07	0	0	4
10	0	3	5
11	0	5	6
12	0	7	7
13	0	6	7
14	0	5	7
15	0	3	7
16	0	0	7
17	0	0	4

A repeating sequence of 16 words is provided by "disconnecting" the Memory Address input # 7.

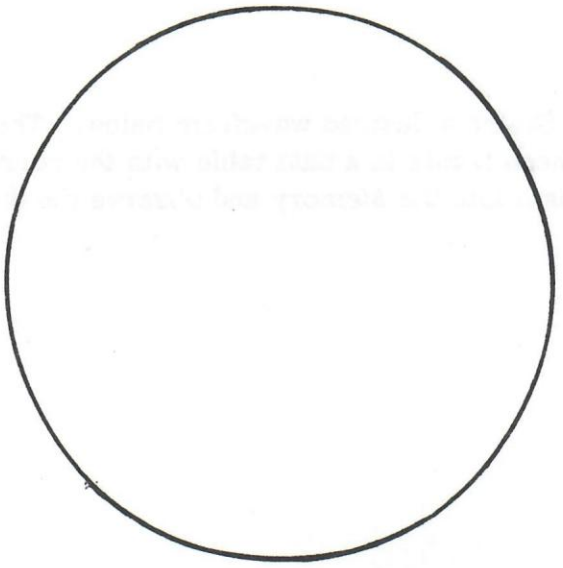
Wire the Logic in 16 - 2. Do not use the external 0.1 mfd components.

Disconnect the CLOCK and keep it disconnected.

Enter the data given in the above table.

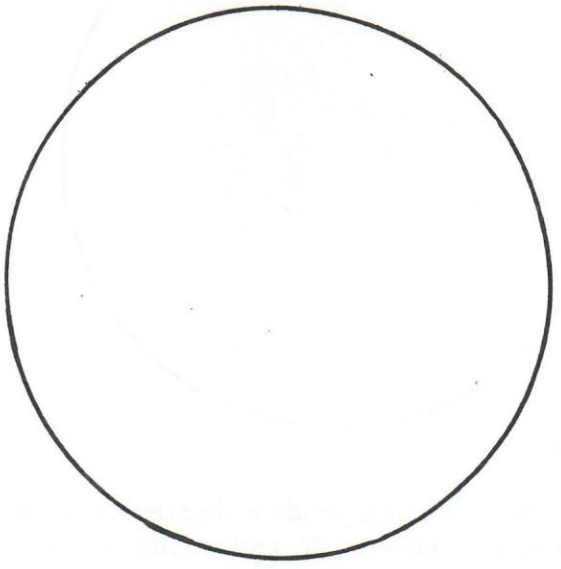
Connect the CLOCK.

Observe the graphical output and sketch below:



POINT BY POINT DISPLAY

The display is a sequence of separate points. Connect the two external 0.1 microfarad capacitors. This provides a "low pass" smoothing filter to yield a "line" between the points. Sketch the new "continuous" waveform below:



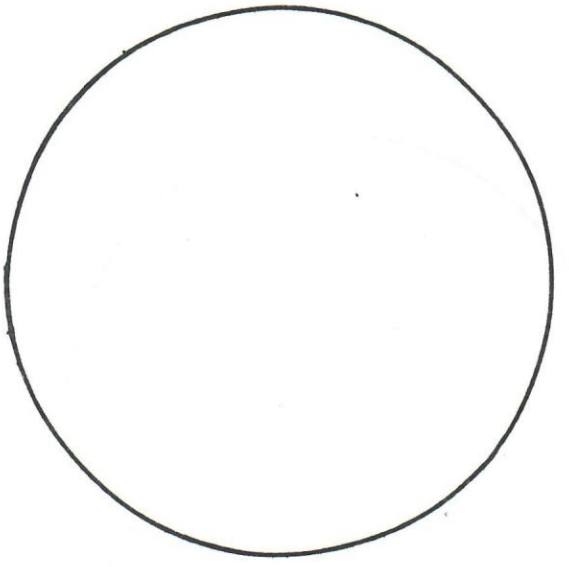
CONTINUOUS DISPLAY

EXERCISE:

1. Sketch a desired waveform below. Then select 32 points on an 8 x 8 grid and list these points in a data table with the corresponding sequential addresses. Enter the data into the Memory and observe the display.

SKETCH DESIRED SHAPE

Plot the resultant graphical display below:



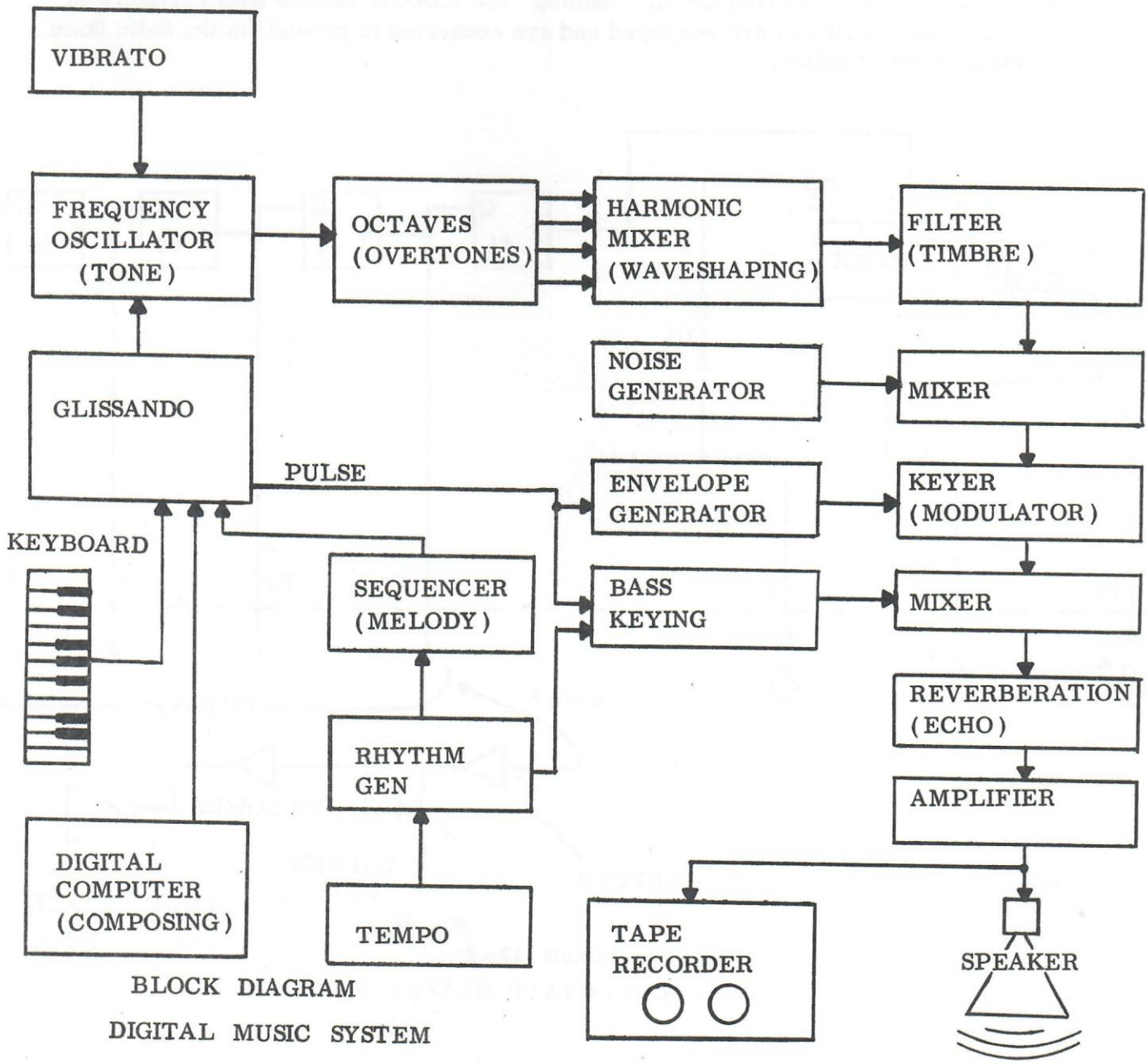
The student can provide a display on a 16 x 16 grid (256 points) by using the full 8 bits with 4 bits for X and 4 bits for Y. The 4 D to A inputs are used and a single resistor of 40 K (to the threshold L) will provide a new least significant bit on the Y output.

OBJECTIVE:

Digital Electronics equipment and techniques are closely related to Music techniques in many respects including frequency variations, tones, timing (beats and rhythm), waveshapes, harmonics and overtones, etc. This experiment shows how the digital and analog computer components are employed in the generation and control of electronic music.

GENERAL:

The general functions of an electronic music system are illustrated in the block diagram below:



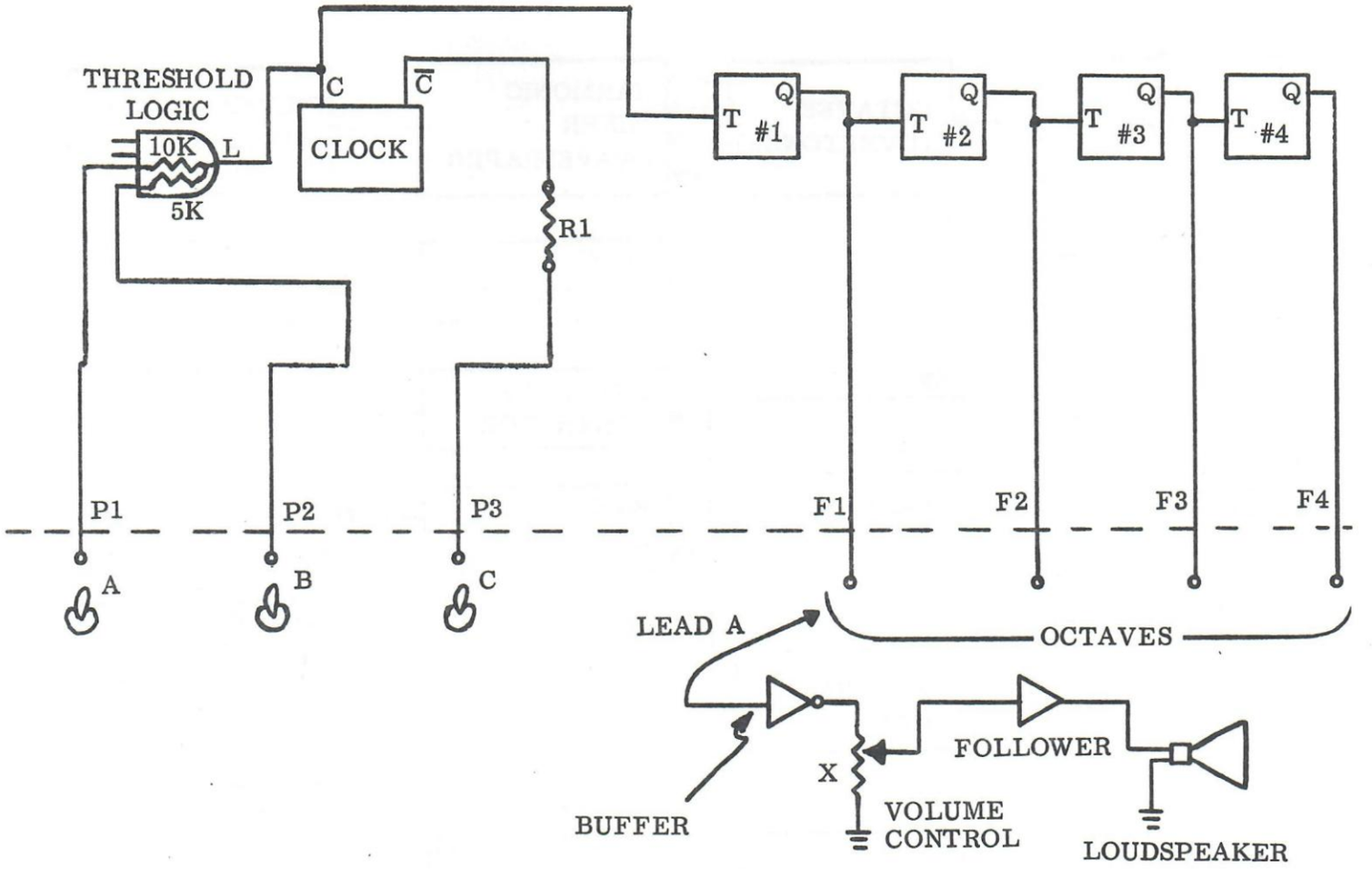
BLOCK DIAGRAM

DIGITAL MUSIC SYSTEM

Each of the different blocks has associated controls. Typically; the Vibrato has a speed or rate, amplitude, on-off control; the Envelope Generator has an attack, decay, sustain control; the Harmonic Mixing and Filters have a variety of overtone mixing amplitude control; etc. The advantage of digital implementation and control is that a digital computer or controller can be connected to the digital electronic music system to automatically modify or arrange, generate or compose, and operate or perform. This experiment demonstrates the operation, implementation and combination of the various digital electronic music components.

DESCRIPTION (TONE VARIATION):

Music tone or pitch correspond to different frequencies that can be obtained from the Ed - Lab CLOCK set at the HIGH frequency position. Variations in frequency or tone are obtained by "loading" the CLOCK outputs with resistors to ground. Fixed resistors are employed and are connected to ground via the Solid State switches as shown below:



Experiment 17 - 1
TONE AND OCTAVE GENERATION

The CLOCK is varied using a 10 Kilohms and 5 Kilohms resistor (Threshold) on one output and a 3 Kilohms resistor on the other output. The clock drives Flip-Flop (#1). Switches A, B, and C provide different combinations and corresponding different tones. Flip-Flops #2, #3, and #4 are connected as a counter and thus divides the basic frequency (F1) by 2, 4, and 8 providing FREQUENCY OCTAVE VARIATIONS. A NOT element acts as a "buffer" for "isolation" and the FOLLOWER drives a SPEAKER directly. Only a loudspeaker is required as an add-on to the Ed-Lab 700 for this application.

PROCEDURE (TONE VARIATION):

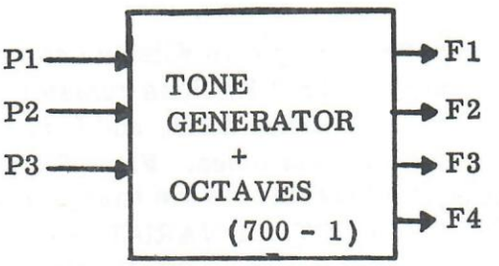
Wire the logic in Figure 17-1. Potentiometer X is used as a volume control. Connect "lead A" to F1 and vary the frequency using Switches A, B, and C. Listen to all 8 combinations in a binary coded sequence as given below:

STEPS	A	B	C
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Using any selected tone, move Lead A to F2, F3, and F4 and observe the OCTAVE variations in tones:

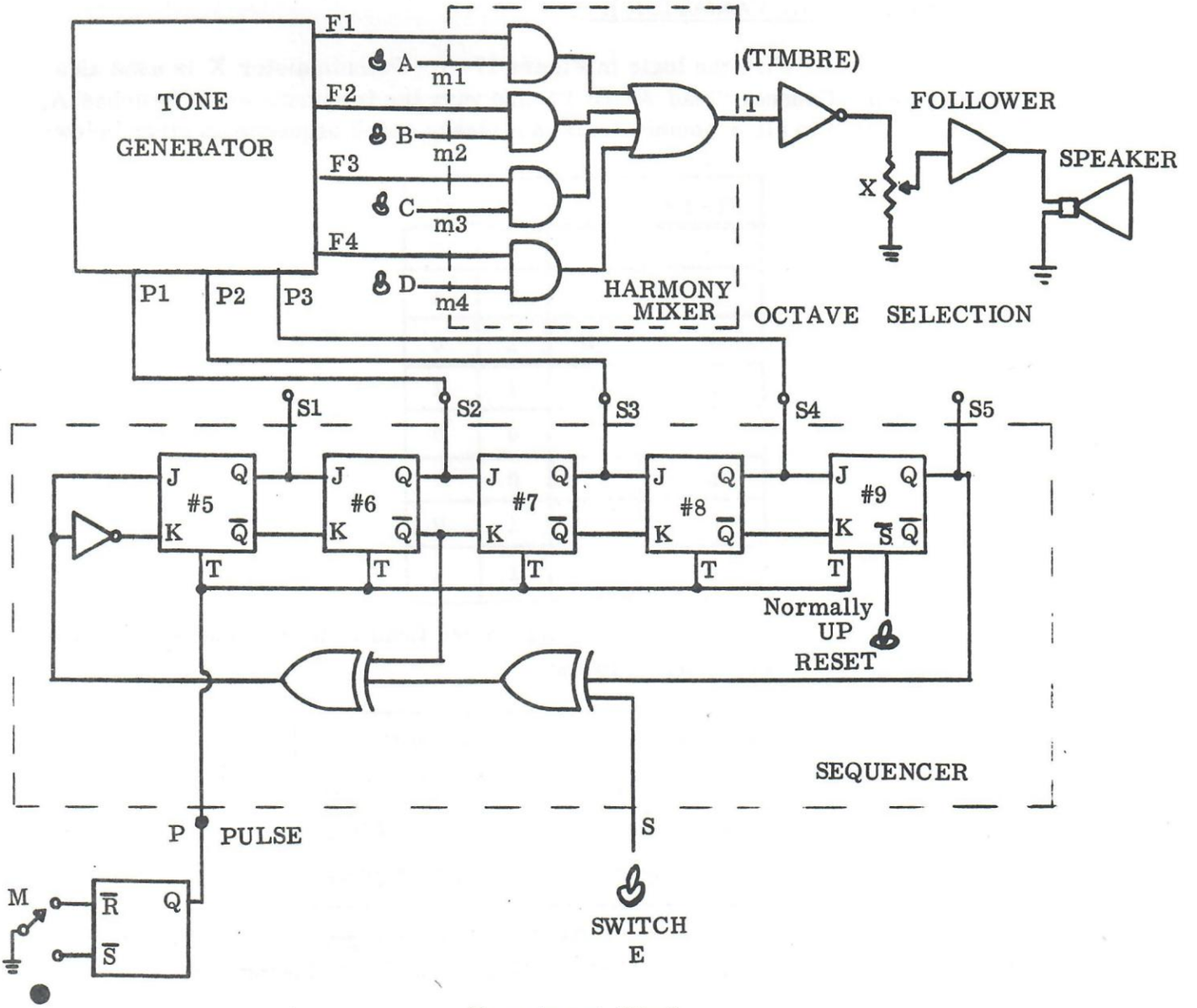
F1	Fundamental	BASIC NOTE
F2	2nd HARMONIC	1st OVERTONE
F3	3rd HARMONIC	2nd OVERTONE
F4	4th HARMONIC	3rd OVERTONE

The basic TONE GENERATOR of Figure 17-1 is denoted in further diagrams with the BLOCK DIAGRAM given on the following page:



PROCEDURE (DIGITAL SEQUENCE CONTROL):

The switches correspond to Manual Tone Control. Digital Control is illustrated by the use of a SEQUENCER as shown below:



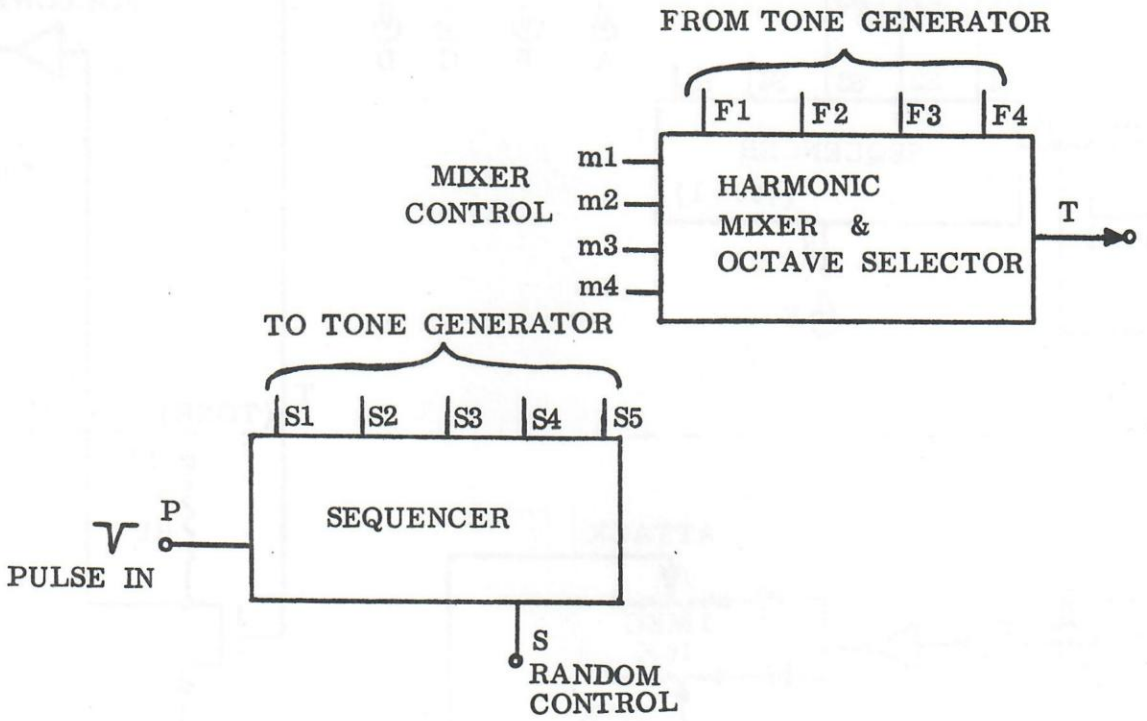
Experiment 17 - 2
DIGITAL SEQUENCE CONTROL

The shift register with feedback steps through all notes in a "pseudo-random manner" and is stepped by Switch M. The octaves are controlled by manual switches and AND gates.

PROCEDURE (DIGITAL SEQUENCER):

Wire the Logic in Figure 17-2 . Step Switch M and listen to the digitally controlled notes and octave sequencing. Switch E changes the random sequence.

The Sequencer and Harmonic (Octave) Mixer logic is drawn in block diagram form as shown below:

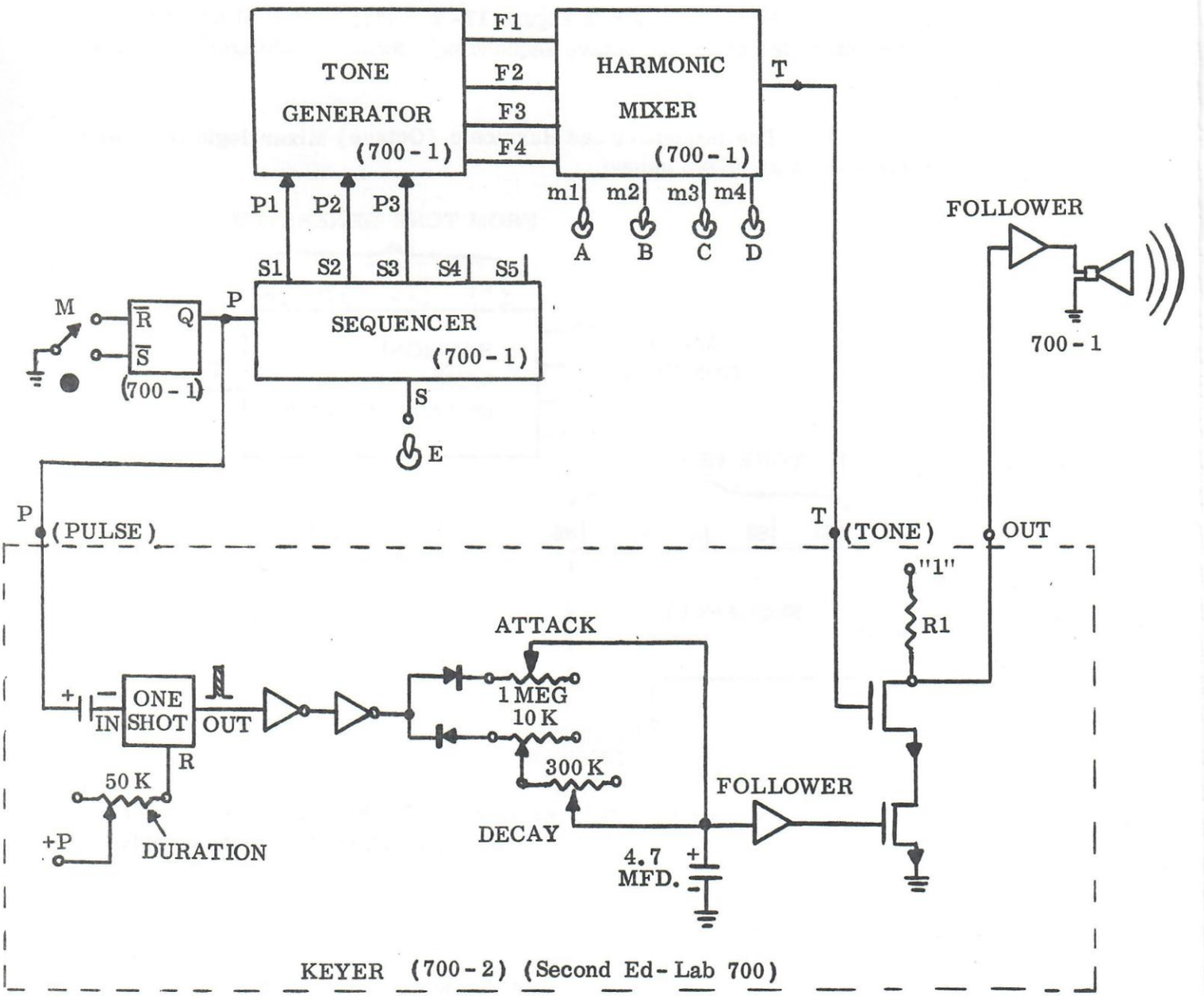


The sequence of notes or MELODY has been determined by a shift register. Better or more musical melodies are achieved by more complex sequencers as described later.

DESCRIPTION (KEYING):

When tones are produced by beating a drum, plucking a string, blowing on a reed, bowing a string, etc., there is a start and stop "modulation" on the tone. This modulation has an "attack" and "decay" shape with a sharp attack for plucking or beating; a slow attack for bowing or blowing; a slow decay for plucking or bowing; and a sharp decay for beating.

The electronic modulation is called KEYING and is performed using a second Ed - Lab 700 and the solid - state switches and one - shot as shown below:



Experiment Logic 17-3
KEYING SYSTEM

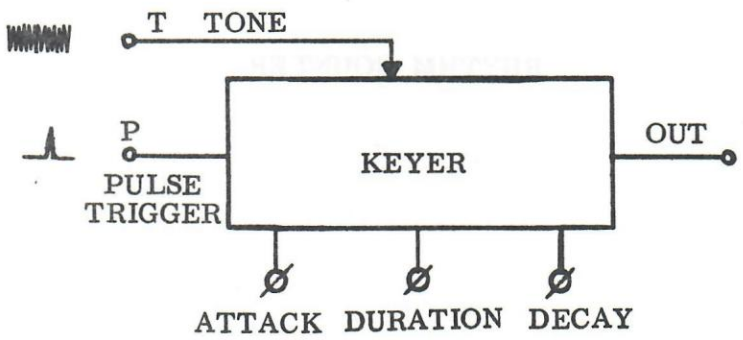
The two solid-state switches are connected in "cascade" with the high frequency tones at one input and the one-shot circuit at the bottom input to modulate the signal ON and OFF. Two resistor - capacitor time constantly are used to adjust attack and decay using three external potentiometers, and an external capacitor as shown. The ONE - SHOT is triggered via another capacitor to yield short pulses. Another external potentiometer controls the width or DURATION of the Keying pulse.

PROCEDURE:

Wire the Keyer on another Ed - Lab 700 together with the remainder of the system. Press Switch M and listen to the Keyed sound. Vary the Pulse Duration, and Attack and Decay time. Comment on the results in the table below:

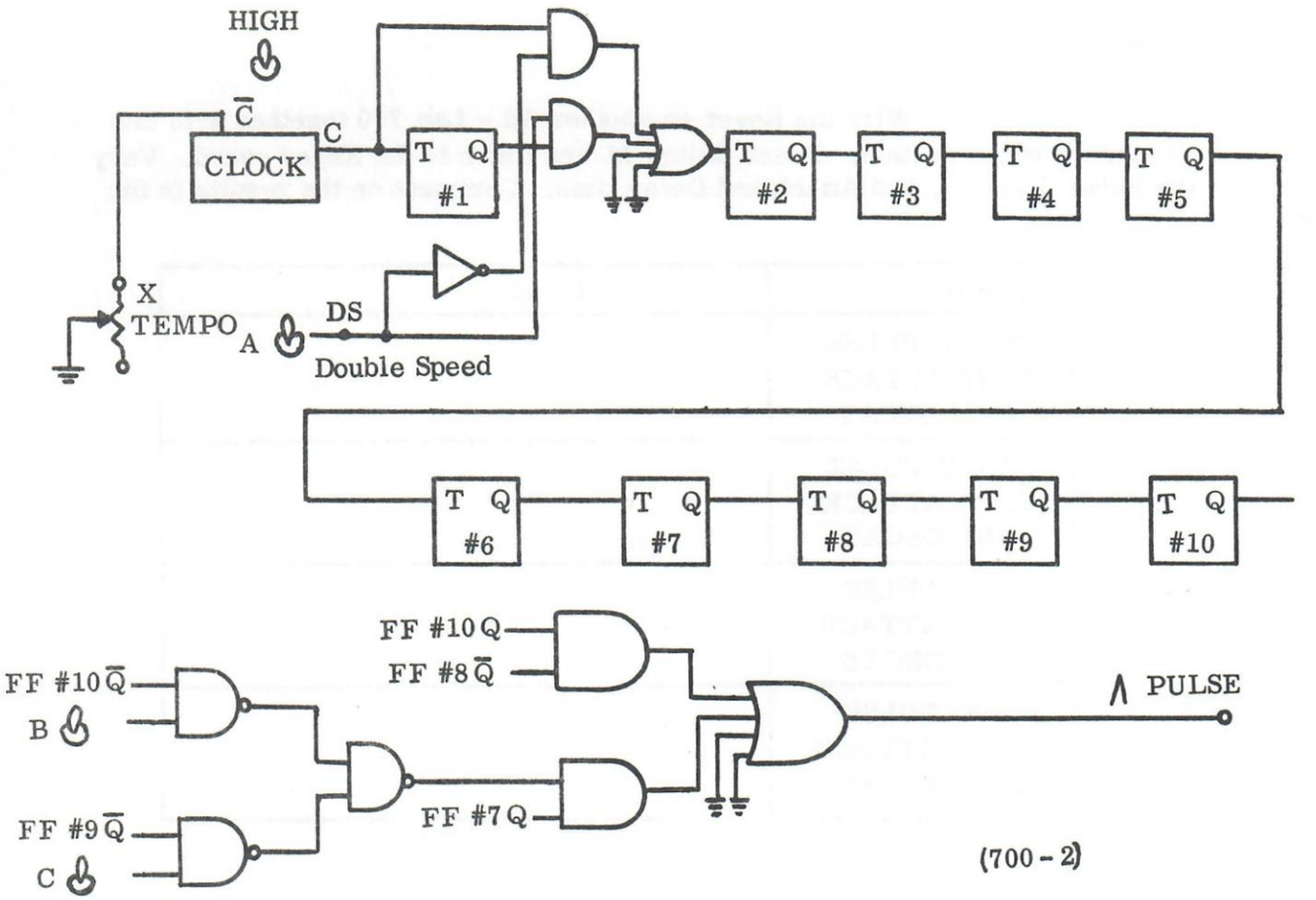
Setting	Sound
SHORT PULSE FAST ATTACK LONG DECAY	
SHORT PULSE FAST ATTACK FAST DECAY	
LONG PULSE SLOW ATTACK FAST DECAY	
LONG PULSE SLOW ATTACK LONG DECAY	

The Keyer block is indicated symbolically below:



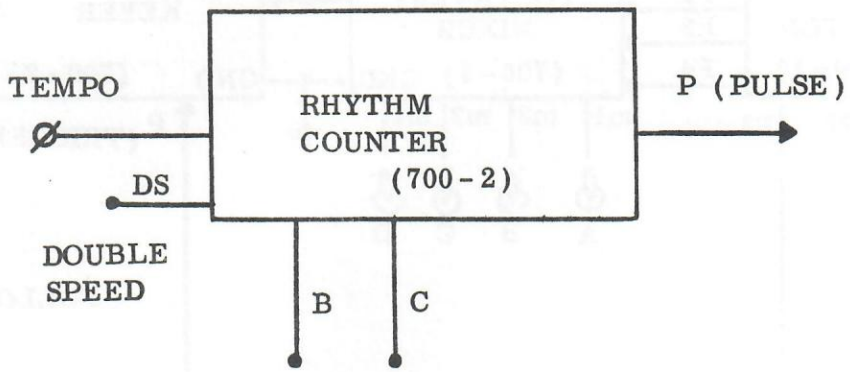
DESCRIPTION (BEATS AND RHYTHM):

The Beat is provided by low frequency timing and the Rhythmic variations are enabled by the decoding of a counter. The second Ed - Lab 700 also provides the Rhythm section. The logic is shown below:



RHYTHM COUNTER

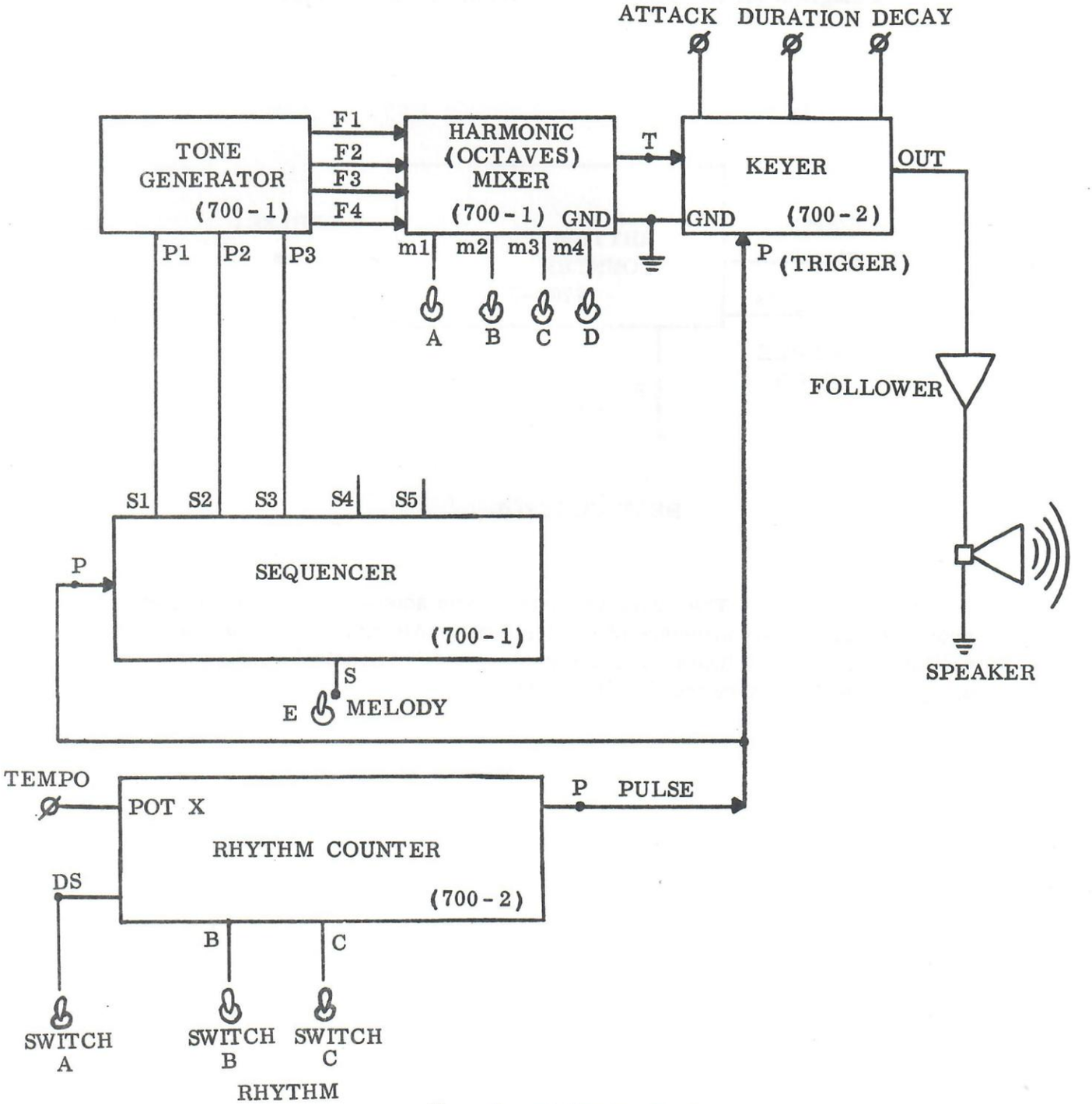
The high frequency Clock is "counted down" to provide the counter bits for slow or rapid beats. The logic shown is for one selected beat with Switches B and C modifying the beat or rhythm. The rhythm pulses will trigger the Sequencer and the Keyer One-Shot. A double speed rhythm is achieved by eliminating Flip - Flop #1 in the counter with the gating shown in the above figure. The block diagram for the RHYTHM COUNTER is shown below:



BEAT CONTROL

The Rhythm Counter Logic shown on the previous page provides a particular sequence of rhythm beats. Alternate logic configurations will provide different beats. A more generalized beat control is given later using the DECODER on the Ed - Lab 900.

Wire the Rhythm Counter on the second Ed - Lab 700 and connect the units as shown below:



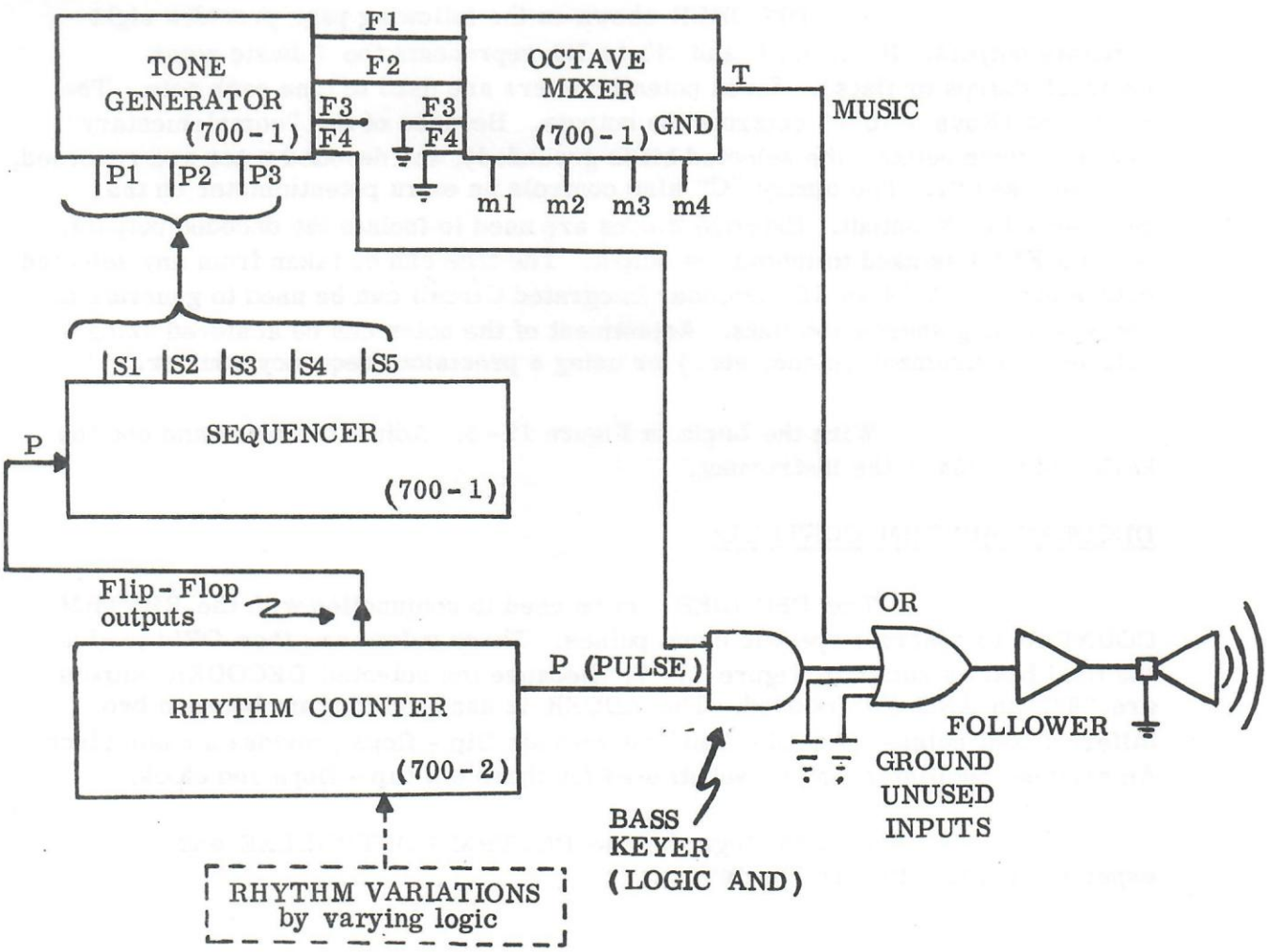
Experiment Logic 17-4

RHYTHM SYSTEM

Run the system varying POT X (Unit #2) to alter the timing and using Switches B and C (#2) to vary the rhythm and Switch E (Unit #1) to vary the melody. Try other rhythm logic variations and vary the attack, duration and decay.

PROCEDURE (BASS RHYTHM):

In the above system, the rhythm pulse affected the playing or sequencing of the basic note. An alternate system is to have a constant melody beat pulse and to have the variable rhythm pulse affect only the bass or lowest octave keying and "add" or "mix" this with the melody section. This system is illustrated below:



Experiment Logic 17-5
BASS RHYTHM

Using the above system, the variable Keying is applied to the high notes, and the lowest octave of the melody is "keyed" using an AND gate (with sharp attack and decay) to produce the Bass rhythm.

Modify the system as shown and listen to the results. Vary the rhythm beat with different logic combinations and vary the melody sequencing by interchanging the sequencer outputs.

DECODER TUNE CONTROL:

Improved precision and versatility in musical performance can be achieved by using the DECODER and external potentiometers. Enhanced musical results requires tuning for the various notes, ease in control of rhythm beats, and ease in selection of note sequencing or melody composition.

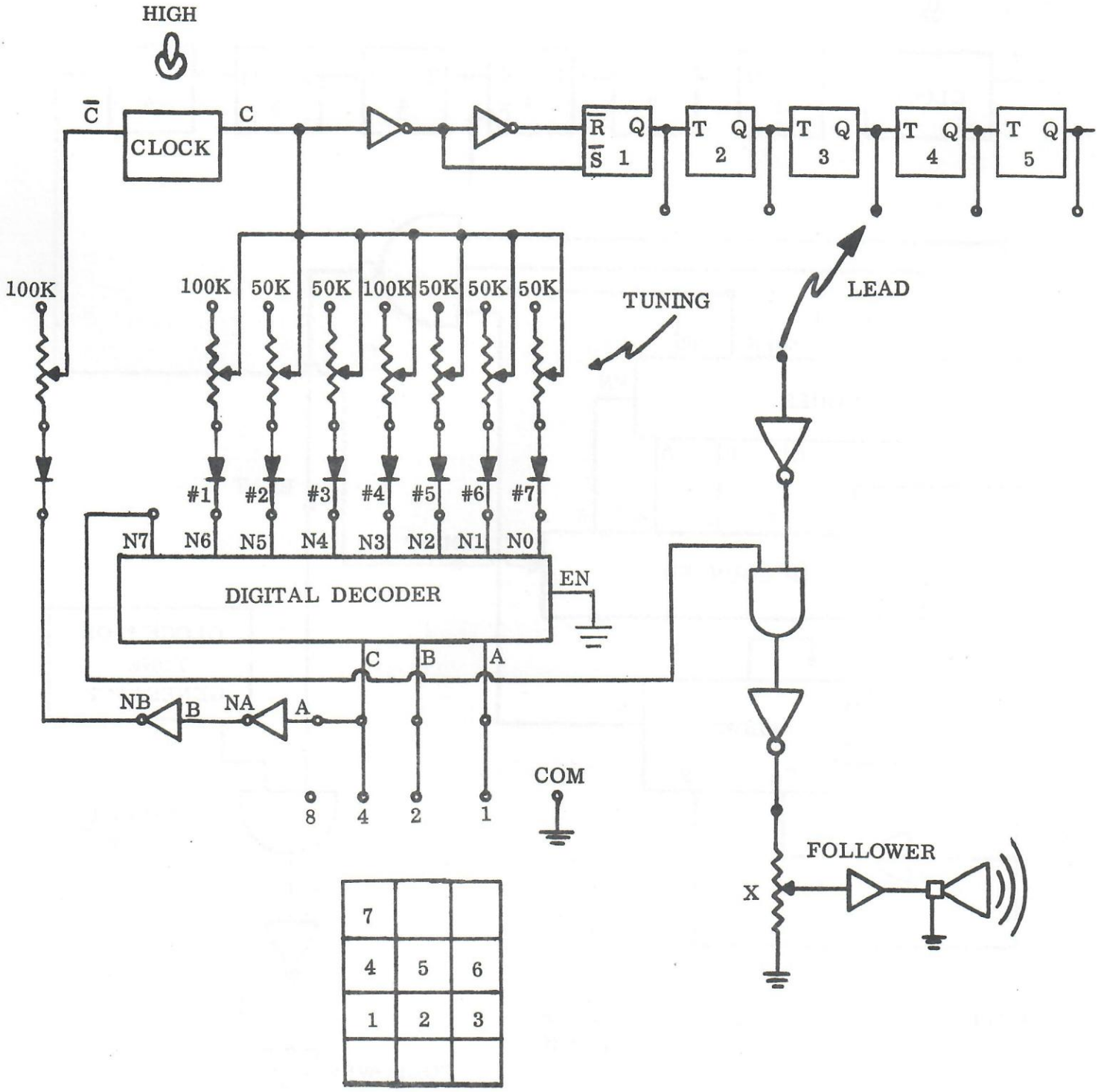
The DECODER shown on the following page provides eight separate outputs. N7 is OFF and N0 to N6 represents the 7 basic notes (without sharps or flats). Seven potentiometers are used to tune each note. The Keyboard (Keys 1 to 7) controls the outputs. Because of the "complementary" keyboard code output (the selected bit is grounded), the decoded notes are reversed, i.e. N6 = key #1. The binary "C" also controls an extra potentiometer on the reverse CLOCK output. External diodes are used to isolate the decoded outputs. N7 (no KEY) is used to inhibit the output. The tone can be taken from any selected octave output. A "4 to 16" Decoder Integrated Circuit can be used to generate all notes including sharps and flats. Adjustment of the notes can be achieved using a reference instrument (piano, etc.) or using a precision frequency counter.

Wire the Logic in Figure 17 - 6. Adjust the notes and use the keyboard to "play" the instrument.

DECODER RHYTHM CONTROL:

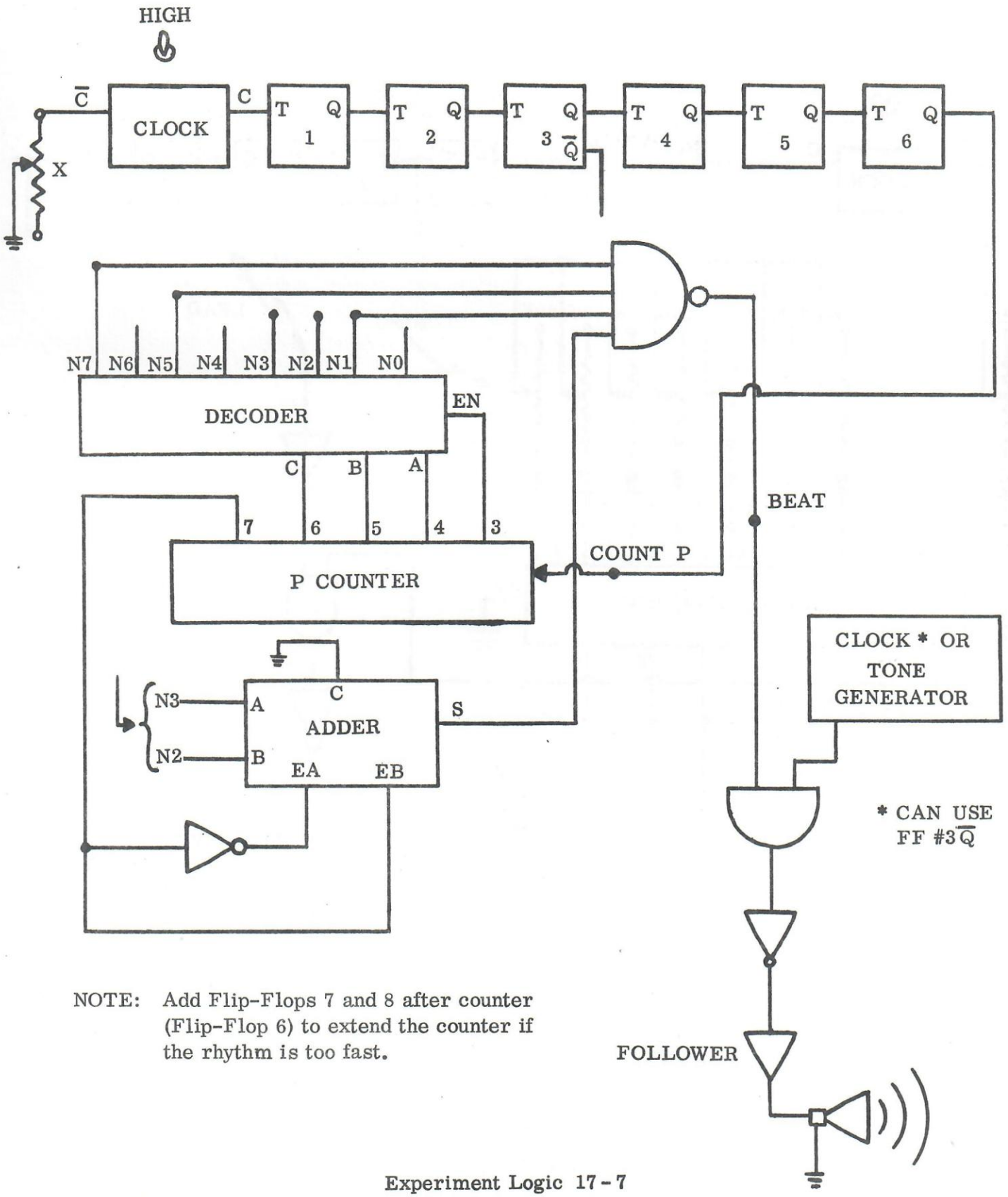
The DECODER can be used in conjunction with the RHYTHM COUNTER to generate specific count pulses. These pulses are then OR'd to give the final beat as shown in Figure 17 - 7. Because the selected DECODER outputs are "0", an AND gate is used. The ADDER is used to alternate between two different beat pulses. An Ed - Lab 700 with six flip - flops provides a basic clock. An external oscillator may be substituted for these six flip - flops and clock.

Wire the logic for the RHYTHM CONTROLLER and experiment with different "beats".



Experiment Logic 17 - 6

TUNING



NOTE: Add Flip-Flops 7 and 8 after counter (Flip-Flop 6) to extend the counter if the rhythm is too fast.

Experiment Logic 17-7

RHYTHM CONTROLLER

COMPOSITION AND MELODY:

The final musical result will depend on the musical talent and initiative of the user. The selection of notes for melody can be achieved using the SEQUENTIAL MEMORY SYSTEM for a preset melody or by using random or rather controlled-random sequencing. Musical organization is based on repetitive patterns using notes that "go well together". The generation of new musical numbers is left as an exercise for the student.

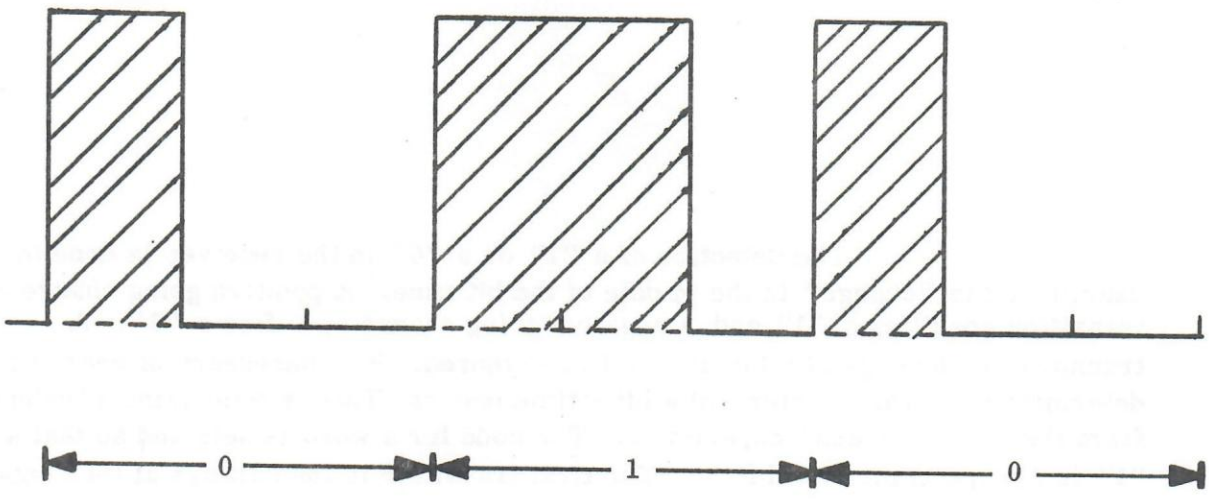
OBJECTIVE:

This experiment demonstrates the high - speed encoding of digital information in a form suitable for transmission or recording using a "phase" modulation and demodulation technique.

DESCRIPTION:

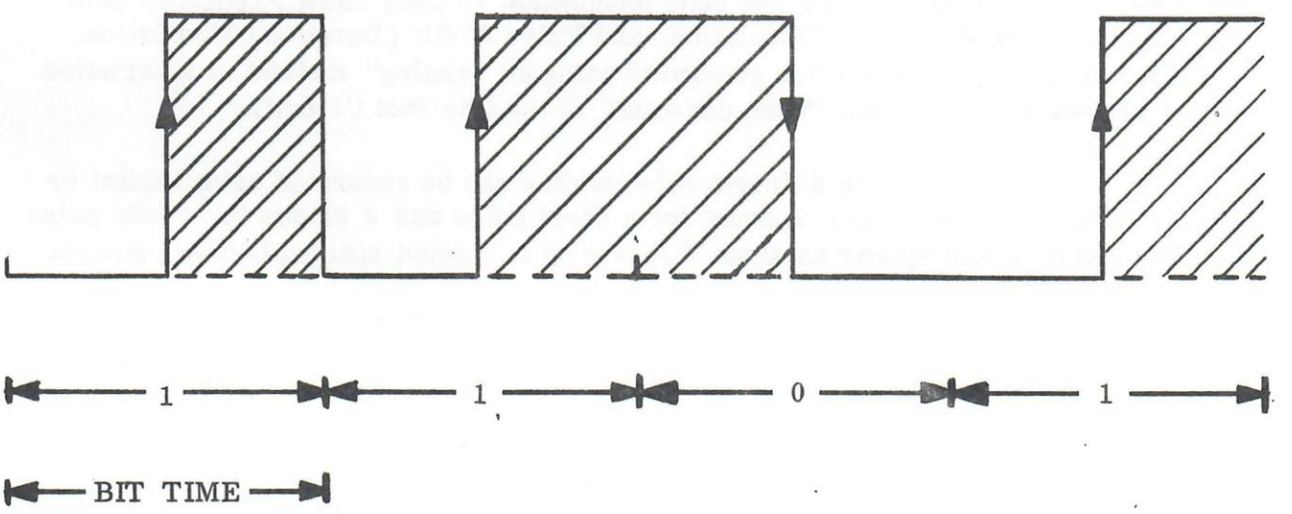
The previous experiment in Digital Communications (Exp. No. 15) described pulse code modulation (PCM) using Frequency Shift Keying, Pulse Amplitude Modulation, and Pulse Width (Duration) Modulation. The variable pulse width was generated using an "analog" method or a variation in resistance to vary the "time constant" of the One Shot Circuit.

The different pulse widths can be generated using digital or counting techniques; e.g. 1 count for a short pulse and 2 counts for a long pulse. The coding would appear as shown below with a 1 count spacing between pulses.



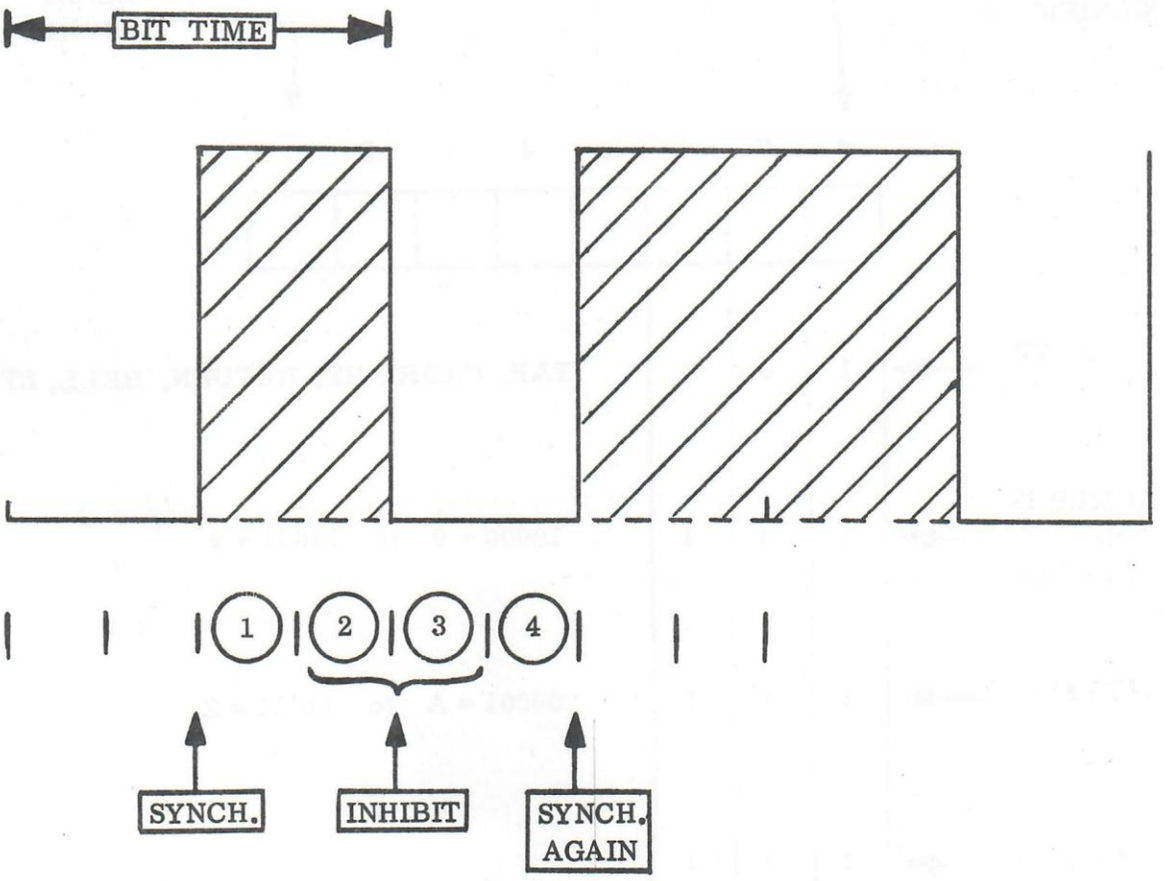
PULSE WIDTH MODULATION

The "phase - modulation" uses only "one - width" pulses with the space(low level)to the left for a "1", and to the right for a "0" as shown below:



PHASE MODULATION

The detection of a "1" or a "0" in the receiver is done by examining the "change" in the middle of the bit time. A positive going change or transition specifies a "1" and a negative going change specifies a "0". A transition at the edge of a bit time is to be ignored. It is necessary of course to determine when the "center" of a bit - time occurs. This is done using a technique from the "data refresh" experiment. The code for a word is selected so that a "1" is always transmitted first. The first transition is then always at the center of a "bit - time". An internal receiver clock at approximately the same rate as the transmitter clock is "synchronized" with a center pulse and counts 4 phases as shown on the following page.

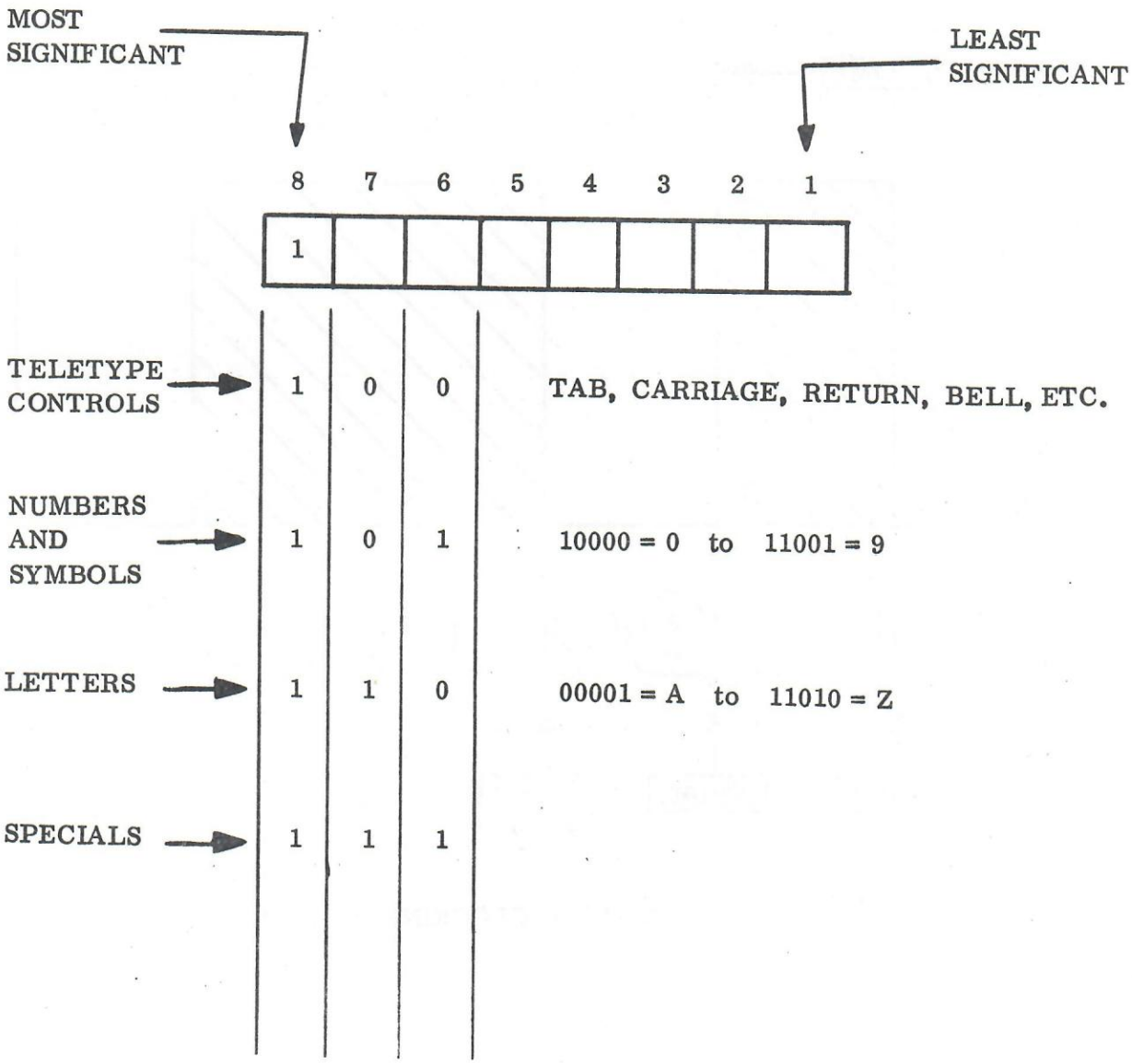


RECEIVER CLOCKING

Any transition occuring during phases 2 and 3 are ignored. When phase 4 is reached, the clock is halted until a new transition re-synchronizes the clock starting at phase 1.

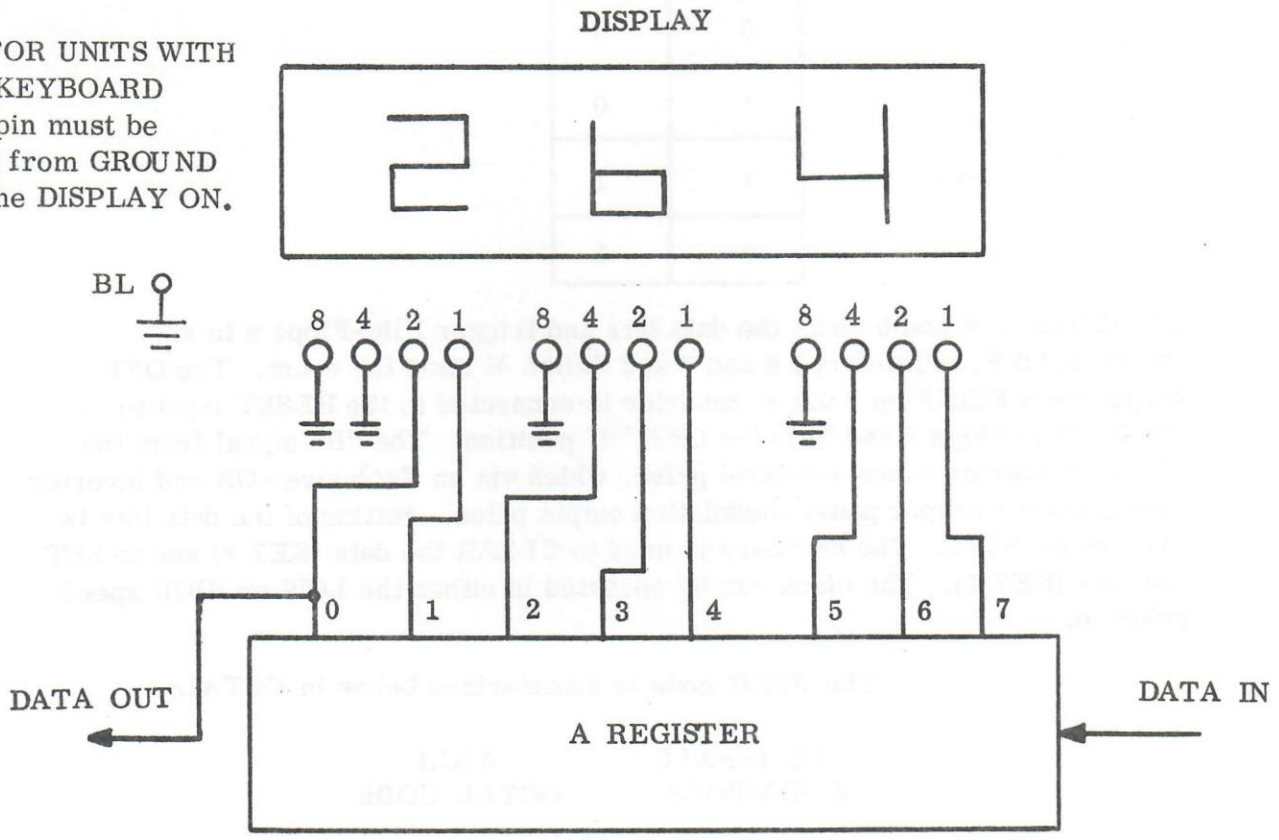
The standard 8-bit word is used in the modem. The eight bit word structure has been adapted as the standard in information data transfer and interchange in the code denoted as the ASCII Code (American Standard Code for Information Interchange).

The ASCII 8-bit code is indicated below:



It is noted that the most significant digit is always a "1". Transmitting the most significant bit first in our system provides the "1" signal for synchronization as required in our modem. (It is recalled that in the normal computer calculations that we have used, the least significant bit is transferred first.) It is thus necessary to place the most significant bit in the shift register "0" position and the least significant bit in the shift register "7" position because the data shifts out of the "0". The display connection is shown below:

NOTE: FOR UNITS WITH 12 KEY KEYBOARD The BL pin must be removed from GROUND to turn the DISPLAY ON.



DISPLAY CONNECTION

The above diagram shows the register outputs as reversed. In wiring the logic on the unit, the leads from the "A" Register to the display have to be connected in a reverse or "criss - cross" manner.

The Transmitter Logic is shown on the following page. An Ed - Lab 900 and an Ed - Lab 700 are used. The phase counter uses Flip - Flops 1 and 2 connected as a "Johnson Counter" Shift Register to generate the four phases with outputs A and B as tabulated below:

PHASE COUNTER

A	B
0	0
1	0
1	1
0	1

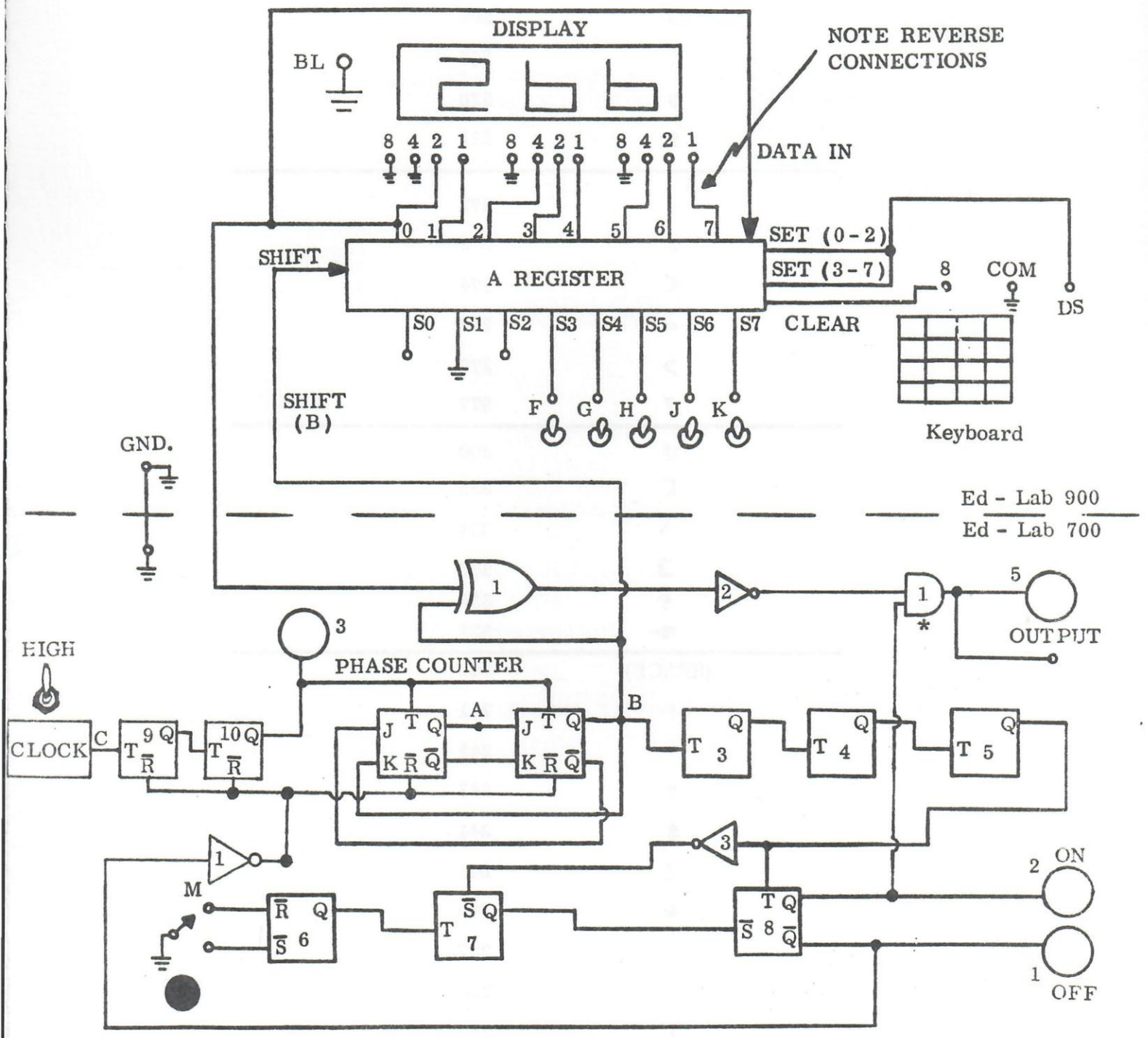
Flip-Flops 3, 4 and 5 count the data bits and trigger Flip-Flops 8 to shut the clock OFF. Flip-Flops 6 and 7 and Switch M start the count. The OFF output from Flip-Flop 8 via an Inverter is connected to the RESET input to "lock" Flip-Flops 9 and 10 in the OFF "0" position. The "B" signal from the phase counter provides a control pulse, which via an Exclusive - OR and Inverter generates the proper phase modulation output pulse. Setting of the data bits is via five switches. The keyboard is used to CLEAR the data (KEY 8) and to SET the data (KEY 1). The clock can be operated in either the LOW or HIGH speed position.

The ASCII code is summarized below in OCTAL:

NUMERALS & SYMBOLS	ASCII OCTAL CODE
0	260
1	261
2	262
3	263
4	264
5	265

(continued on page 8)

NOTE: FOR UNITS WITH
12 KEY KEYBOARD
The BL pin must be
removed from GROUND
to turn the DISPLAY ON.



Experiment 18 - 1

PHASE - MODULATION TRANSMITTER

*Added to prevent transmission of data when setting new data into transmitter keyboard.

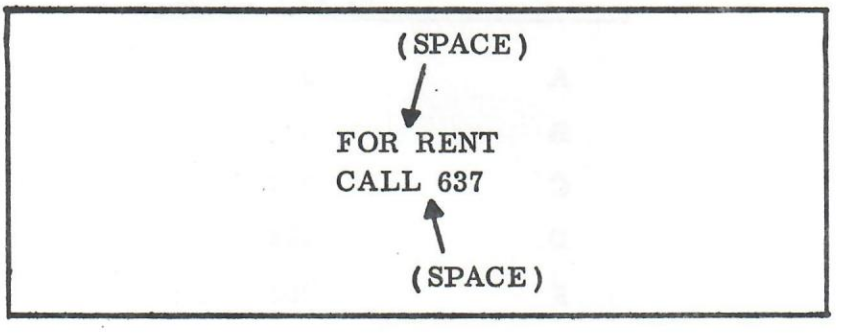
NUMERALS & SYMBOLS	ASCII OCTAL CODE
6	266
7	267
8	270
9	271
:	272
;	273
<	274
=	275
>	276
?	277
@	300
⊏	333
\	334
⊐	335
↑	336
←	337
(SPACE)	240
!	241
"	242
#	243
$	244
%	245
&	246
'	247
(	250
)	251
*	252
+	253
,	254

NUMERALS & SYMBOLS	ASCII OCTAL CODE
—	255
.	256
/	257
(Bell)	207
(Line feed)	212
(Carriage Return)	215
(Start of message)	201
(End of message)	203

LETTERS	ASCII OCTAL CODE
A	301
B	302
C	303
D	304
E	305
F	306
G	307
H	310
I	311
J	312
K	313
L	314
M	315
N	316
O	317
P	320
Q	321

LETTERS	ASCII OCTAL CODE
R	322
S	323
T	324
U	325
V	326
W	327
X	330
Y	331
Z	332

A typical "message" is:



The "coded" message is

WORD #	SYMBOL	OCTAL
0	F	306
1	O	317
2	R	322
3	(space)	240
4	R	322
5	E	305
6	N	316
7	T	324
10	(Line feed)	212
11	(Carriage-return)	215 (con't)

WORD #	SYMBOL	OCTAL
12	C	303
13	A	301
14	L	314
15	L	314
16	(space)	240
17	6	266
20	3	263
21	7	267

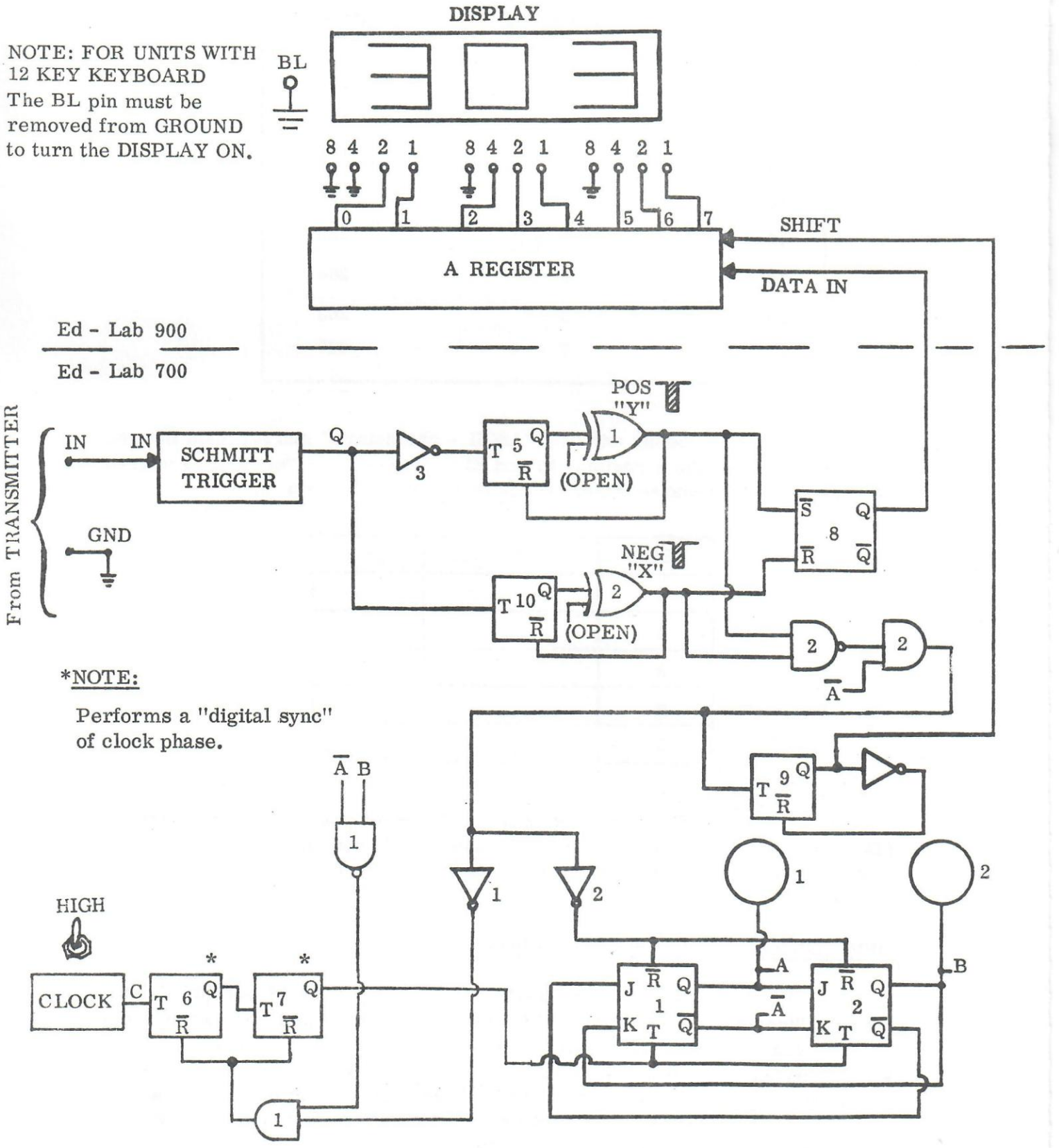
Select any five "ASCII - characters" and set them into the register. Connect the A register S1 and S2 to "ground" or "open" as required. S0 is always open because the most significant bit is always a "1".

TEST	OCTAL NO.	✔
1		
2		
3		
4		
5		

With the clock in the LOW speed position, transmit the data (Depress Switch M, and observe the transfer). Do not remove any wires from this experiment. The receiver experiment follows using another Ed - Lab 700 and 900.

DESCRIPTION (Phase - Modulation Receiver):

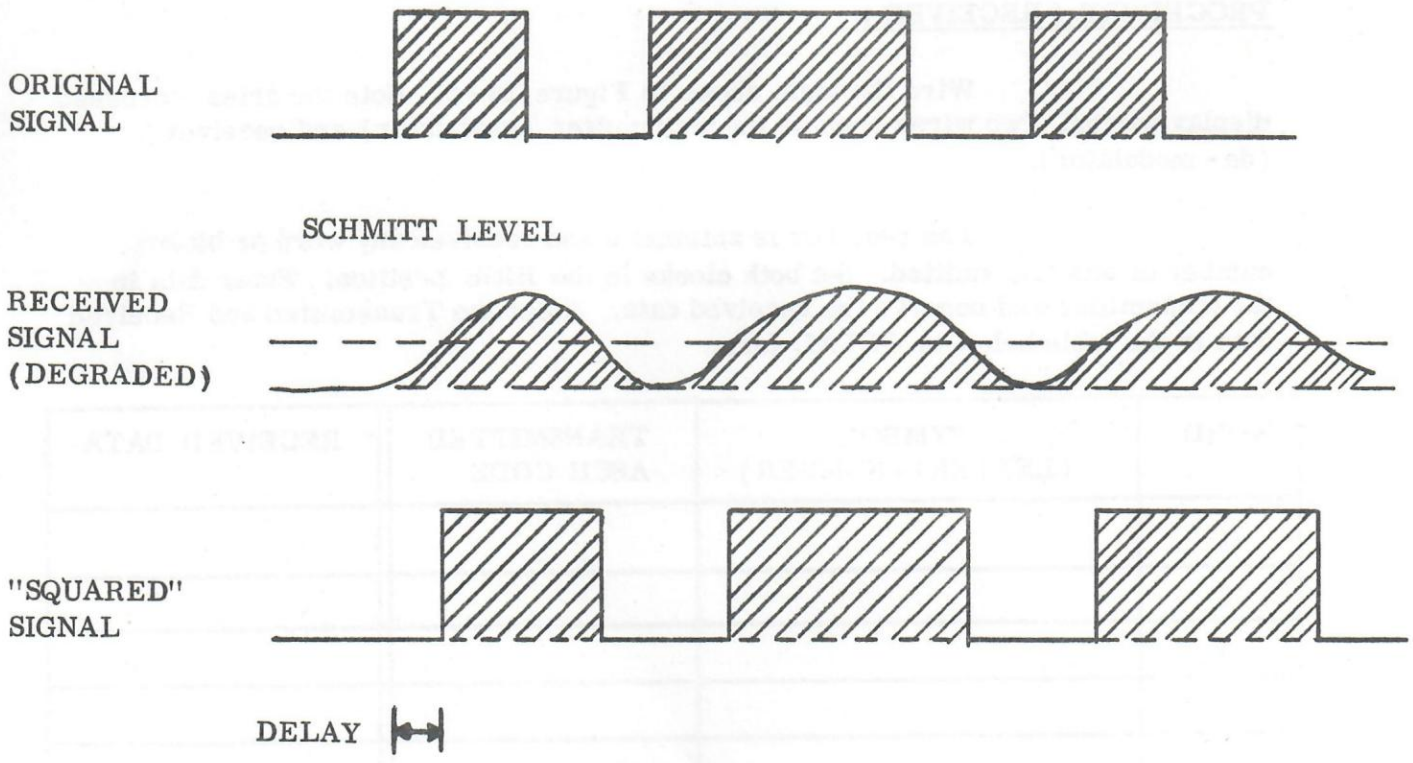
The receiver must generate a signal which permits "viewing" the data "transition" and which inhibits viewing the transition in-between bits. Such a signal can be called a "data window". The receiver requires a CLOCK whose frequency is "near" the frequency of the transmitter. The exact frequency is not required because the CLOCK is "re-synchronized" at each data transition. The logic for the Phase Modulator Receiver is shown on the following page and also uses an Ed - Lab 900 and an Ed - Lab 700.



Experiment 18 - 2

PHASE - MODULATION RECEIVER

The input signal is applied to the SCHMITT TRIGGER which yields a "square" pulse from the received input signal which may have been "degraded" by the limitation in "bandwidth" in the communications link. For the Phase Modulated Signal, a typical degraded signal is shown below:



"SQUARED" SIGNAL

The "Squared Signal" is "delayed" from the original signal but this causes no problem in the "recovery" of the data.

The squared signal triggers the "one-shot" logic of Flip-Flop #10 and EX-OR #2. This is triggered by a "negative transition" and thus "pulse X" specifies a "0" bit transition. An inverter (#3) triggers the one-shot with Flip = Flop #5, and pulse Y specifies a "1" bit. Pulses X and Y set and clear Flip-Flop #8 which has the correct data for input to the Shift Register (on the Ed-Lab 900). The Flip-Flops #1 and #2 are connected as a Four Phase Shift Register and will automatically stop at A = 0, B = 1 via NAND #1.

The first pulse will always be a "data" pulse for a "1" bit of data because of the code used. NAND #2 and AND #2, will generate a negative pulse when the first data bit is received, AND $\overline{A} = 1$. The first pulse also resets the Phase Register to AB = 00 and the CLOCK Flip-Flops 6 and 7 are enabled to shift the phase register. The register step is AB = 10 and AB = 11

producing an $\overline{A} = 0$ which inhibits (AND #2) a shift pulse or reset on the undesired transition. Three pulses later, the condition is $A = 0$ and $\overline{A} = 1$ and the CLOCK is stopped. The next transition will be for valid data and will shift data into the "A" Register and re-start the phase counter and clock.

PROCEDURE (RECEIVER):

Wire the logic shown in Figure 18 - 2. Note the criss - crossed display wiring. Two wires connect the transmitter (modulator) and receiver (de - modulator).

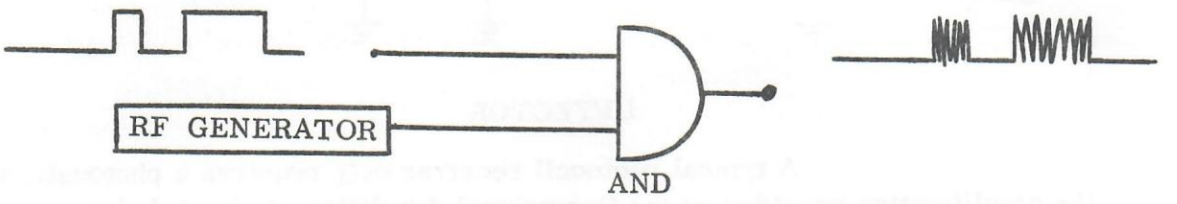
The receiver is automatic and receives any word or bit or number of bits transmitted. Set both clocks in the HIGH position. Enter data in the transmitter and observe the received data. Enter the Transmitted and Received data in the table below in OCTAL form:

WORD	SYMBOL (LETTER) (NUMBER)	TRANSMITTED ASCII CODE	RECEIVED DATA
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			
17			
18			
19			
20			

EXERCISE:

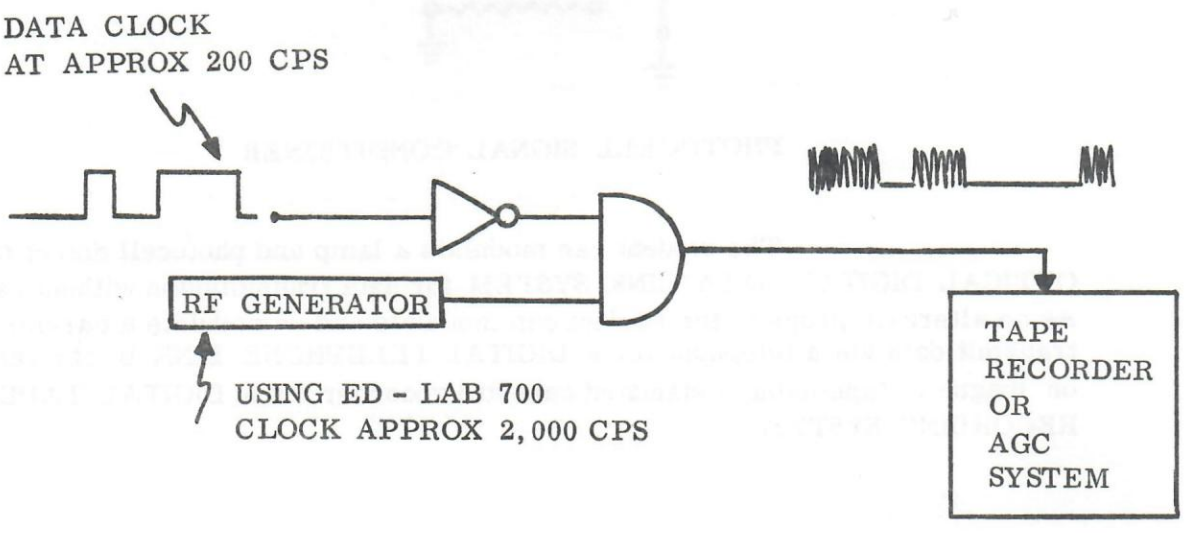
1. The Digital Modem signal can be "modulated" for transmission via the telephone cables, an optical link, a tape recorder, etc.

Modulation can be performed using a carrier frequency and an AND gate as shown below:



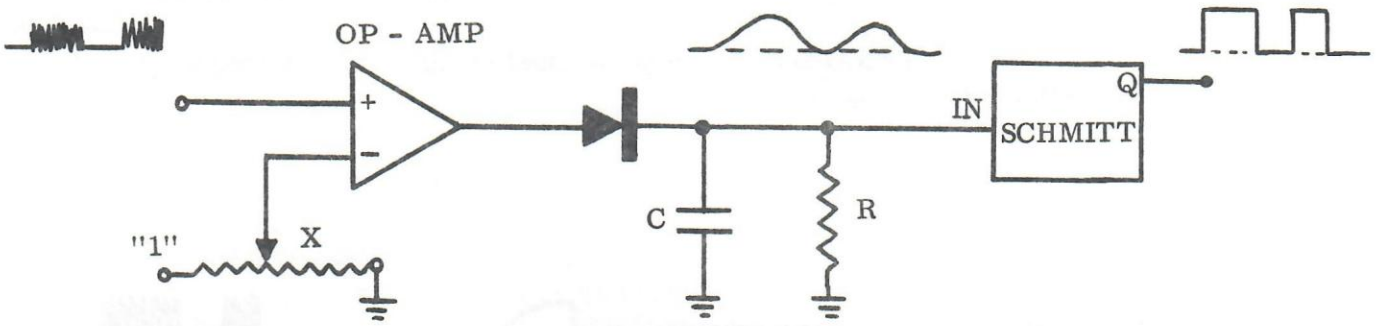
PCM CARRIER FREQUENCY MODULATOR

A "detector" is then also required. When using a system with AUTOMATIC GAIN CONTROL (AGC), such as with some cassette recorders, the carrier should then be ON most of the time. This could be accomplished with an Inverter as shown below:



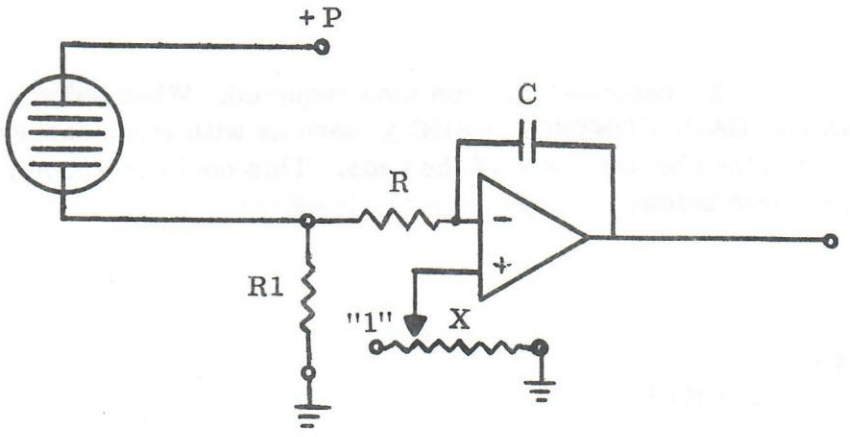
TAPE RECORDER OR TELEPHONE MODULATOR

The detector for the amplitude modulated signal is shown below:



DETECTOR

A typical photocell receiver only requires a photocell, with the amplification provided by the Operational Amplifier as shown below:



PHOTOCELL SIGNAL CONDITIONER

The student can modulate a lamp and photocell detect for an OPTICAL DIGITAL DATA LINK SYSTEM for data transmission without cables. As an alternate project, the student can modulate and demodulate a carrier and transmit data via a telephone for a DIGITAL TELEPHONE LINK or for recording on magnetic tape using a standard cassette recorder for a DIGITAL TAPE RECORDING SYSTEM.

OBJECTIVE:

Prior experiments have demonstrated programming techniques for arithmetic and other logical applications. Computers are used to perform many different types of operations and this experiment explores special techniques used when solving a variety of problems.

GENERAL:

In addition to the computer's usage in solving arithmetic problems, other general uses include the processing of data or information such as:

RETRIEVAL	:	Obtaining data from a specific location.
LOCATION	:	To determine the address of some specific data.
SEARCH	:	To determine the location and value of typically the lowest number.
SORT	:	To place data in an order, typically in increasing or alphabetical order.

Other arithmetic applications would include PROBLEM SOLVING, SCALING, DOUBLE PRECISION, FLOATING POINTS, REAL - TIME TIMING, as well as the use of computational "ALGORITHMS".

Descriptions and examples are given in this experiment.

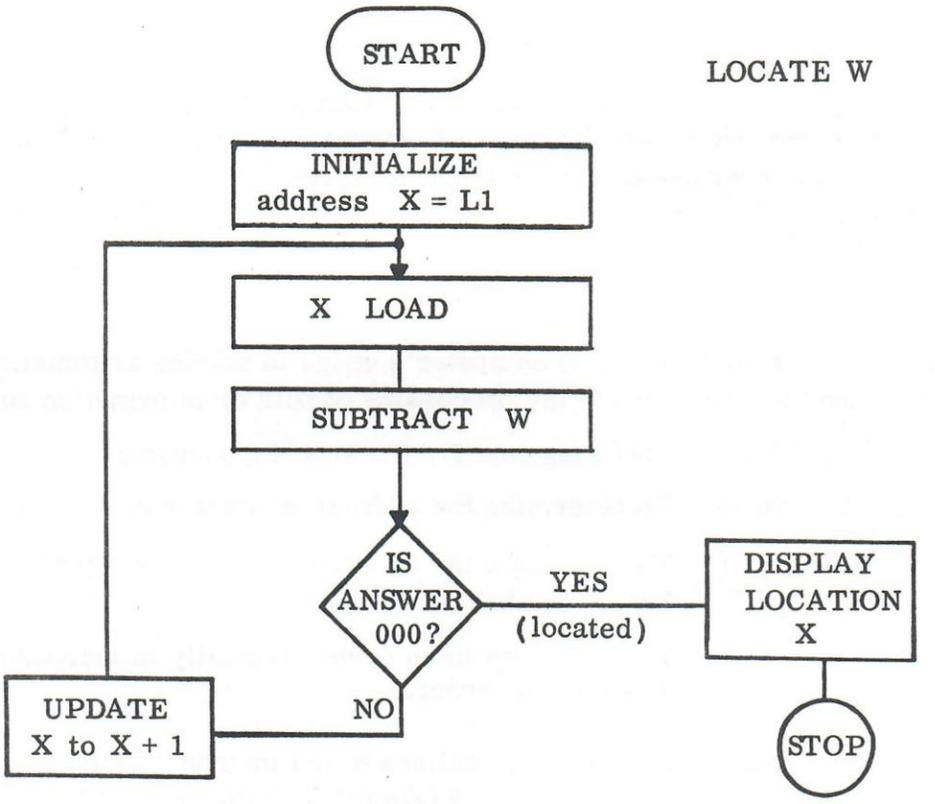
RETRIEVAL:

The retrieval of data from location 27 is obtained simply by INSTRUCTION:

27 LOAD

LOCATION:

In order to locate a specific item or word of data, we have to examine each item and look for a "match". This is done by subtraction and looking for a 00 difference. If data is located in addresses starting at L1, the flow chart for the LOCATION PROGRAM would be as shown on the following page:



FLOW CHART

LOCATION

SEARCH:

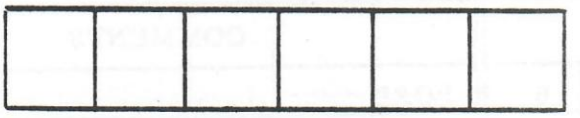
A lowest number can be detected by subtracting the number under examination from the previous lowest number. If the answer is "positive", then we have examined a new lowest number. If the answer is zero (000), then both are the lowest and equal. The location is given by the address of the number under examination.

SORT:

Consider the following numbers:

5
7
9
3
2
8

Place these numbers below in increasing order:



One approach is to examine all numbers, SEARCH for the lowest number, and place it in the left box; then to examine all of the remaining numbers for the lowest number, etc.

Although this job is tedious, the computer can perform the operations at a rapid pace. Other approaches are also possible. In assorting a full deck of cards in order; we would select one card, then place the next card in the proper relative position (lower or higher), then take the next card (number) and scan (examine) the cards until we reach a card location which is too high, etc. This approach should take less operations. The set of rules that are used to solve a particular problem is called an ALGORITHM. Typical examples are the rules or Algorithms for multiplication and division.

PROCEDURE (LOCATION PROBLEM):

A Location Problem is now stated:

Ten Data words are located in memory addresses 03 to 14 (octal). It is required to determine the lowest or first location of a particular data word or to have an indication that the word is not in memory.

The Program is given on the following page.

EXERCISE:

1. It is left as an exercise for the student to prepare the Flow Chart for this problem. The Coded Program can be used as a guide.

Enter the program into the computer without any data in addresses 00 to 14.

Select a sequence of ten data words for entry in memory 03 to 14 and select a word to be "located" for entry in memory position 02. Word 01 is always 141 for transfer to the first instruction (15). Word 00 must be entered each time and specifies the address of the first data word to be examined.

Ed-Lab 900 : PROGRAM CODING FORM

PROGRAM :		LOCATION			
ADDRESS		WORD			COMMENTS
0	0	0	3	6	LOAD
0	1	1	4	1	TRA TO (15)
0	2				LOCATION DATA #Y
0	3				DATA 1
0	4				2
0	5				3
0	6				4
0	7				5 DATA
1	0				6
1	1				7
1	2				8
1	3				9
1	4				10
1	5	0	2	4	SUB # Y
1	6	2	0	0	TON TO (21)
1	7	3	6	4	SUB # 001
2	0	3	0	0	TON TO (31)
2	1	0	0	6	LOAD (previous address)
2	2	3	5	5	ADD (for new address)
2	3	0	0	2	STORE
2	4	3	4	5	ADD
2	5	2	6	0	TON (27) NEG COMPLETED
2	6	3	7	1	TRA (00) POS CONTINUE
2	7	3	6	6	LOAD
3	0	3	7	7	HALT (NOT FOUND, DISPLAY 001)
3	1	0	0	6	LOAD (ADDRESS)
3	2	3	7	7	HALT (FOUND, DISPLAY (000) ADDRESS)
3	3	ø	ø	ø	-
3	4	0	2	2	CONSTANT
3	5	0	1	0	CONSTANT
3	6	0	0	1	CONSTANT # 001
3	7	0	0	0	CONSTANT # 000

An example is given below using the first letter from the first ten words of the Declaration of Independence and coded in ASCII.

ADDRESS	DATA	WORD
00	-	036
01	-	141
02		
03	WHEN	327
04	IN	311
05	THE	324
06	COURSE	303
07	OF	317
10	HUMAN	310
11	EVENTS	305
12	IT	311
13	BECOMES	302
14	NECESSARY	316

(ADDRESS) LOAD

(STEP 15) TRANSFER

SELECT LETTER TO BE LOCATED

Enter the data in addresses 00 to 14. With the computer in AUTO, start the operation and observe the results. Enter the results in the table below:

TEST	WORD TO BE LOCATED (02)		LOCATION
1	HUMAN	310	
2	IN	311	
3	DISOLVE	304	
4	IT	311	
5			
6			
7			

Comment on the results.

QUESTION:

2. How can the operator search for the word IT (311) in the presence of IN (311) in an earlier location?

SCALING:

In actual computations, the octal numbers have dimensions such as FEET, YARDS, POUNDS, OUNCES, DOLLARS, MILES PER HOUR, CENTS PER POUND, etc. The actual number is related to the computer octal number by a SCALE FACTOR including a DIMENSION. Thus, the octal number 135 with a decimal scale factor (SF) of SF = 150 dollars, represents the quantity as calculated below:

OCTAL NUMBER $\boxed{135}_{8}$

DECIMAL EQUIVALENT $(1 \times 64) + (3 \times 8) + (5 \times 1)$

$\boxed{93}_{10}$

QUANTITY = 093 x 150 dollars

Quantity = \$13,950

THEREFORE

$\boxed{135}_{8} \longrightarrow \boxed{\$13,950.}$

Frequently, the scale factor is written in a manner that requires division of the computer number by the scale factor (e.g. SF = 1/150 dollars). Other scale factors refer to the number notation read in a fractional form. We will employ the scale factor as used in the above example.

When a number is to be inserted into the computer, a scaling division is required. Thus, the following procedure is employed:

INPUT DOLLARS \$6,953.00

DIVIDE BY SCALE FACTOR

$$\frac{\$6,953}{150 \text{ dollars}} = 46\ \frac{13}{150} \text{ (decimal)}$$

The remainder is eliminated resulting in a ROUND-OFF ERROR of $\frac{13}{150}$,

DECIMAL = [46]$_{10}$

= (0 x 64) + (5 x 8) + (6 x 1)

OCTAL NUMBER = [056]$_{8}$ + ROUND OFF ERROR

THUS: [$6.953] ⟶ [056]$_{8}$ + ROUND-OFF ERROR

When adding or subtracting, the scale factors of both numbers must be equal. When multiplying, the scale factor of the product is the "product of the two scale factors".

An example is given below:

RATE x WEIGHT = COST

85 cents per pound x 1.4 pounds = 1190 cents or $11.90

In selecting a scale factor, the maximum and minimum values must be considered, with a resulting round-off error.

Consider the scale factor of:

1. SF = 1 cent
2. SF = 2 ounces

The SF for rate must then be given by

SF (cents) = SF (ounces) x SF (cents per ounce)

$$\text{SF (cents per ounce)} = \frac{\text{SF (cents)}}{\text{SF (ounces)}}$$

= 1/2 cents per ounce

The maximum octal digit is 377 corresponding to decimal $255_{(10)}$.

The maximum numbers are then:

MAX. COST = 255 x 1 = 255 cents or $2.55

MAX. WEIGHT = 255 x 2 = 510 ounces or 31 pounds, 14 oz.

MAX. RATE = 255 x 1/2 = 127.5 cents per ounce or $20.40 per pound

An example is now given:

The cost of a certain item is $1.50 per pound. The weight of a particular purchase is 1 pound, 4 ounces. What is the total cost?

COST = 1.25 pounds x $1.50 per pound

= $1.87 + 1/2 Cent REMAINDER

The computer calculations are as follows:

$$\text{SCALED RATE} = \frac{150 \text{ cents}}{16 \text{ ounces}} \times \frac{}{1/2 \text{ cents per ounce}} = 18\frac{3}{4} \text{ (decimal)}$$

$$(\text{round-off}) = 19 \text{ (decimal)}$$

$$\text{RATE} = \boxed{023}_{\text{OCTAL}}$$

$$\text{SCALED WEIGHT} = 20 \text{ ounces} \times \frac{1}{2 \text{ OUNCES}} = 10 \text{ (decimal)}$$

$$\text{WEIGHT} = \boxed{012}_{\text{OCTAL}}$$

The calculated answer for total is $T = R \times W$:

where: T = total cost
R = rate
W = weight

The computer calculation is :

	023	OCTAL		019	DECIMAL
x	012	OCTAL	x	010	DECIMAL
=	276	OCTAL	=	190	DECIMAL

The answer is thus:

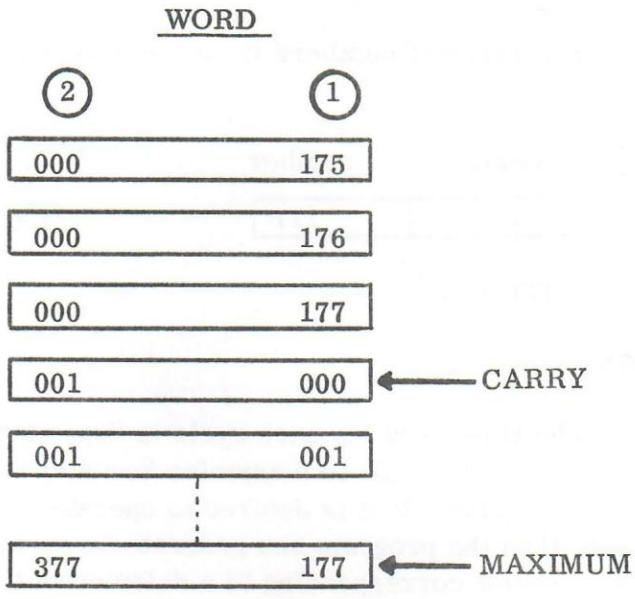
276 CENTS in OCTAL
or 190 CENTS in DECIMAL
or $1.90

The error (answer is $1.87 + 1/2 CENT) is due to the round - off from $18\frac{3}{4}$ to 19 for the SCALED RATE. The increase by 1/4 resulted in a product error of 1/4 x 10 or $2\frac{1}{2}$ CENTS. This error is naturally called a ROUND - OFF ERROR. The programmer must decide what round - off is acceptable and scale the problem accordingly. Note that in the actual answer of $1.87 + 1/2 cent, there is also a round - off error. The customer must be charged $1.87 or $1.88 because the "cash - resolution" is 1 cent. A different scaling would improve the resolution but will result in a lower "maximum" for the required data.

The maximum can be "doubled" by using two computer numbers to represent a single quantity in what is referred to as DOUBLE PRECISION.

DOUBLE PRECISION:

Consider a scaled dollar number with SF = 1 CENT. The maximum quantity that can be stored in a single word is 255 CENTS or 377 CENTS in octal. This can be substantially increased using two words. Consider a positive octal number with a maximum of 177 . An extra 1 yields a 200 which can be detected (TRN) to add a 1 into another register. The counting would be as illustrated below:



The maximum is now equivalent to 8 + 7 bits or 15 bits or 32,767 counts scaled as $327.67. This is a sizable increase from $2.55 prior to using double precision.

EXERCISE:

3. Consider the problem with six numbers stored in locations 01 to 06. It is desired to add these numbers using DOUBLE PRECISION to prevent overflow and provide an accurate answer. Prepare the Flow Chart, Symbolic Program, and Coded Program. Select various numbers and run a test problem on the computer to "debugg" the program.

4. Describe a concept for TRIPLE PRECISION.

5. Describe the use of DOUBLE PRECISION in multiplication.

FLOATING POINTS:

The double precision was used to increase the range of the 8 bit number which normally ranges from 001 or 000 to 377 (octal). The use of two words, where one represents a number and the other represents an "exponent of 2", would also increase the range. This is illustrated below:

	exponent	number		
	005	035	=	035×2^{005}
(2's complement negative number)	373	035	=	035×2^{-005}

The range of numbers in now enormous. The maximum positive number is now

exponent	number
177	177

This is: 177×2^{177}

REAL - TIME TIMING:

The time used for each cycle is dependent on the basic clock frequency. A calculation can be made to determine how many instructions are required to achieve a given time duration. If it is desired to operate some external equipment at 10 second intervals, then the program can generate an output after a preset number of instructions are performed corresponding to a calculation time of 10 seconds. The program given below subtracts 001 from the number stored in (06) until 200 (negative number) is reached resulting in a HALT. Instructions (00) and (04) are performed once. Instructions (01) and (02) are performed for each subtraction.

	ADDRESS	INSTRUCTION	COMMENT
	00	06 (6)	LOAD (06)
	01	07 (4)	SUB (001)
	02	03 (4)	TON
	03	00 (1)	TRA
	04	05 (7)	HALT
	05	000	CONSTANT
TO SET TIMING →	06	100	TIME SET
	07	001	CONSTANT

Enter the program into the computer. Start the program at (00) and use a wristwatch to measure the time from START to HALT.

Enter the time below:

N COUNT	TIME (SECONDS)
100	

Using the above results, estimate the time for various "counts" and the count for various "times". Enter the data in the table below; and run the program with different "N COUNTS" in (06); and measure the elapsed times.

	CALCULATED		MEASURED
TEST	N COUNT IN (06)	TIME (SECONDS)	ELAPSED TIME (SECONDS)
1	100		
2		60	
3		22	
4	153		
5	001 (minimum)		
6	177 (maximum)		
7			

Compare the calculated and measured results.

EXERCISE:

6. The detection of a completed subtraction is determined by a "negative" number. Thus, the maximum N count is 177. We can double this by using a maximum count of 377 and detecting a "000" number. Prepare the flow chart and coded program for the expanded timing system which detects "000".

SQUARE - ROOT ALGORITHM:

An ALGORITHM is a set of rules or procedures or steps followed to solve a problem. There may be many different algorithms that can be used to solve a particular problem. The trick is to find a simple one. There are three common algorithms for multiplication of a many digit number. The product of 18 x 12 can be solved using procedure (1): adding 18 a total of 12 times, or procedure (2): multiplying 18 x 2 and then adding 18 x 1 shifted one place to the left, or procedure (3); adding 18 a total of 2 times, and then adding 18 only 1 time but shifted one place to the left. The examples are shown below:

PROBLEM 18 x 12

Procedure 1:

```
  18
  18
  18
  18
  18
  18
  18
  18
  18
  18
+ 18
 216
```

Procedure 2:

```
  18
x 12
  36
+ 18
 216
```

Procedure 3:

```
  18
  18
+ 18
 216
```

When an adding machine(without automatic multiplication) is available, procedure 3 is used as the best algorithm. The procedure 2 algorithm is used with most problems. And we often use the procedure 1 algorithm when multiplying by small numbers. Consider the problem $6,937 x 2.

Procedure 2:

```
 6,937
x     2
$13,874
```

Procedure 1:

```
  6,937
+ 6,937
$13,874
```

What algorithm would you use to find the square - root of N? Various methods are used and they all generally require "guessing" at a solution and then making subsequent better guesses. A common computer algorithm uses NEWTON's METHOD. This method starts with guessing a solution (G). Divide the number (N) by (G) and obtain a quotient (N/G). If (N/G) = G, then we have made a proper guess and $\sqrt{N} = G$ or G x G = N. If not, a better guess is obtained by:

$$\text{NEW GUESS} = \frac{(N/G + G)}{2}$$

An example is used to find the square root of 25. (We guess at 1).

N = 25

	G	N/G	N/G + G	$\frac{(N/G + G)}{2}$
FIRST	1	25/1 = 25	26	13
SECOND	13	25/13 = 1.9	1.9 + 13	7.5
THIRD	7.5	25/7.5 = 3.3	3.3 + 7.5	5.4
FOURTH	5.4	25/5.4 = 4.6	4.6 + 5.4	5
FIFTH	5	25/5 = 5	5 + 5	5

The program is given on the following page. The division (N/G) is performed by the "successive subtraction" algorithm. The division by 2 is done by a shift to the right. Any remainder (1/2) is shifted into the "sign" position and must be removed (step (33)).

EXERCISE:

7. Prepare the flow chart for the SQUARE - ROOT problem. Refer to the CODED PROGRAM.

Enter the program and select various examples. The program is run in the AUTO mode and automatically HALTS after each guess. Record each new computer guess in a table and depress Switch M for the next computer guess:

Try the following typical example:

(1) FIND $\sqrt{81}$ $81_{(\text{decimal})} = 121_{(\text{octal})}$

(2) We arbitrarily select a first guess of 121 (octal)

$\sqrt{121}$	GUESS
FIRST ENTERED	121
1 st	051
2nd	025
3rd	014
4th	011
5th	011

The answer is 011 (octal) or 9 (decimal), i.e. $\sqrt{81} = 9$.

Ed-Lab 900 : PROGRAM CODING FORM

PROGRAM : SQUARE - ROOT NEWTON'S METHOD $\sqrt{X}$

ADDRESS		WORD			COMMENTS
0	0	D	D	D	ENTER NUMBER X *
0	1	D	D	D	ENTER FIRST GUESS FOR $\sqrt{X}$
0	2	2	4	6	LOAD # 000
0	3	2	5	2	STORE QUOTIENT
0	4	0	0	6	LOAD X
0	5	0	1	4	SUB FIRST GUESS
0	6	1	4	0	TON OVER (TO 15)
0	7	3	6	2	STORE REMAINDER
1	0	2	5	6	LOAD QUOTIENT/PARTIAL
1	1	3	5	5	ADD # 001
1	2	2	5	2	STORE QUOTIENT
1	3	3	6	6	LOAD REMAINDER
1	4	0	4	1	TRA 05
1	5	2	5	6	LOAD QUOTIENT/FINAL
1	6	0	1	5	ADD GUESS
1	7	0	0	3	S RIGHT (DIV BY 2)
2	0	3	2	0	TON (OVERFLOW)
2	1	3	6	2	STORE (AVERAGE)
2	2	2	4	7	HALT
2	3	3	0	1	TRA 31 (TO START AGAIN)
2	4	0	0	0	CONSTANT
2	5	ø	ø	ø	QUOTIENT
2	6	ø	ø	ø	-
2	7	ø	ø	ø	-
3	0	ø	ø	ø	-
3	1	0	1	2	STORE (NEW GUESS)
3	2	0	1	1	TRA 02
3	3	3	7	4	SUB # 200 (OVERFLOW)
3	4	2	0	1	TRA 21
3	5	0	0	1	CONSTANT
3	6	ø	ø	ø	REMAINDER
3	7	2	0	0	CONSTANT

START — HALTS — DIVISION — AVERAGE

* Be sure to use computer in single - step when entering data into memory.
D = DATA. (Use only POS numbers; i.e. not 2 or 3 in first digit)

Enter other examples below:

FIND $\sqrt{N}$	example 1	2	3	4	5	6
N octal	061	173	173			
enter first guess		100	173			
computer 1		040	076			
2		021	037			
3		014	021			
4		013	014			
5		013	013			
6			013			
7						
correct answer	$\sqrt{49} = 7$			$\sqrt{121} = 11$		

It is noted that the answers are calculated to the nearest 001. The maximum N = 177. There are thus limitations in number size and accuracy. The previous discussions on scaling and double precision point the way to improved calculations.

If we wish to find the square - root of 0.36 (decimal), we could use a scale factor of 1/100 decimal for N resulting in a scale factor of 1/10 (decimal) for the answer. For $\sqrt{0.36}$, we would use a computer N of 36; our answer of 6 is then scaled to an answer of 0.6. Our accuracy has now been scaled to 0.1.

PROBLEM SOLVING:

Perhaps the most difficult task in programming is to devise the method to be used in solving a particular problem. Simple addition and subtraction problems are readily formulated. However, complicated word - problems must be "translated" into number - problems in order to be solved. A typical example is given below.

"WHAT IS THE LOWEST TWO DIGIT DECIMAL NUMBER SUCH THAT WHEN THE DIGITS ARE REVERSED, THE NEW NUMBER IS EQUAL TO THE ORIGINAL NUMBER PLUS THE SQUARE OF THE ORIGINAL LEAST SIGNIFICANT OR UNIT DIGIT?"

The answer is 26.

We reverse 26 to obtain the new number 62, then

$$62 = 26 + (6 \times 6)$$

We would not like the idea of starting by trying 11 and 12 and 13, etc. until we reach 26. The computer can use an algorithm which starts at 11 and tries all possibilities. Consider the binary number for 26.

0	0	0	1	1	0	1	0
128	64	32	16	8	4	2	1

= 26

The reversed answer 62 is:

0	0	1	1	1	1	1	0

= 62

But how do we formulate the instructions to provide a binary number which is equal to reversed decimal digits?

One solution is as follows:

Consider the decimal number XY.

A typical number is 29 where X = 2 and Y = 9. The decimal number is defined as N = 10X + Y, or N = 10•2 + 9 which is 29. Multiplication by 10 is multiplication by 8 + 2 or three shifts and one shift to the left in a binary number. The reversed digits are; R = 10Y + X

or R = 10•9 + 2 = 92.

The equation defining our problem is thus:

DECIMAL $\quad YX \stackrel{?}{=} XY + Y \cdot Y$

BINARY $\quad 10Y + X \stackrel{?}{=} 10X + Y + Y \cdot Y$

Consider the correct answer XY = 26.

X = 2 $\qquad$ Y = 6

0 0 1 0 $\qquad$ 0 1 1 0

10Y + X

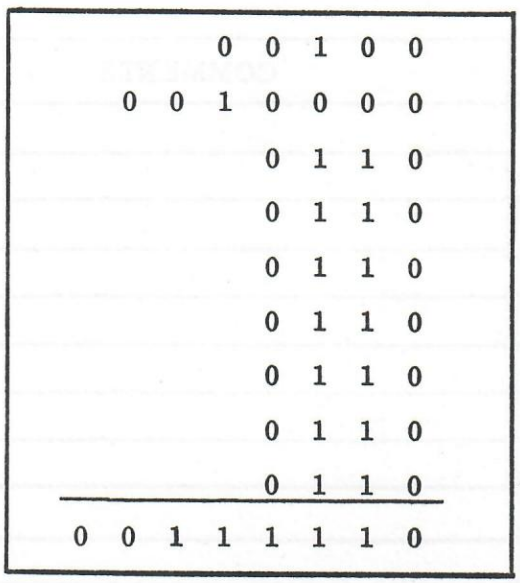
			0	1	1	0	0
		0	1	1	0	0	0
				0	0	1	0
0	0	1	1	1	1	1	0

2Y, 8Y } 10Y

plus X

sum (62 decimal)

10X + Y + Y•Y



			0	0	1	0	0	2X
	0	0	1	0	0	0	0	8X
				0	1	1	0	plus Y
				0	1	1	0	+ Y times Y
				0	1	1	0	
				0	1	1	0	
				0	1	1	0	
				0	1	1	0	
				0	1	1	0	
0	0	1	1	1	1	1	0	sum (62 decimal)

10X

Both SUMS check therefore both sides of the equation are equal and 26 is correct.

EXERCISE:

8. Prepare the flow chart for the solution to the following problem:

WHAT IS THE LOWEST TWO DIGIT DECIMAL NUMBER SUCH THAT WHEN THE DIGITS ARE REVERSED, THE NEW NUMBER IS EQUAL TO TWICE THE ORIGINAL NUMBER PLUS TWO?

A standard Program Coding Form is generally used and provides a uniform program appearance with convenience for the user. A blank is given on the following page.

Ed-Lab 900 : PROGRAM CODING FORM

PROGRAM :

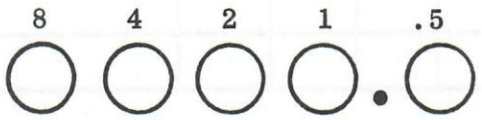
ADDRESS		WORD			COMMENTS
0	0				
0	1				
0	2				
0	3				
0	4				
0	5				
0	6				
0	7				
1	0				
1	1				
1	2				
1	3				
1	4				
1	5				
1	6				
1	7				
2	0				
2	1				
2	2				
2	3				
2	4				
2	5				
2	6				
2	7				
3	0				
3	1				
3	2				
3	3				
3	4				
3	5				
3	6				
3	7				

OBJECTIVE:

This experiment demonstrates how the Computer System can be connected to Input and Output Peripheral Systems such as a Printer, Card Punch, Teletype, Display, etc. The Digital Voltmeter Display system is also demonstrated.

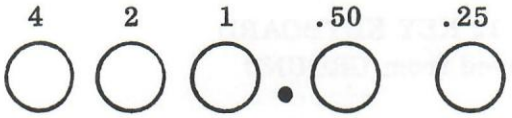
DESCRIPTION (DIGITAL VOLTMETER DISPLAY):

The Digital Voltmeter experiment in the Ed - Lab 700 sequence (Experiment No. 73) results in a binary coded voltage reading with decimal point coded as:



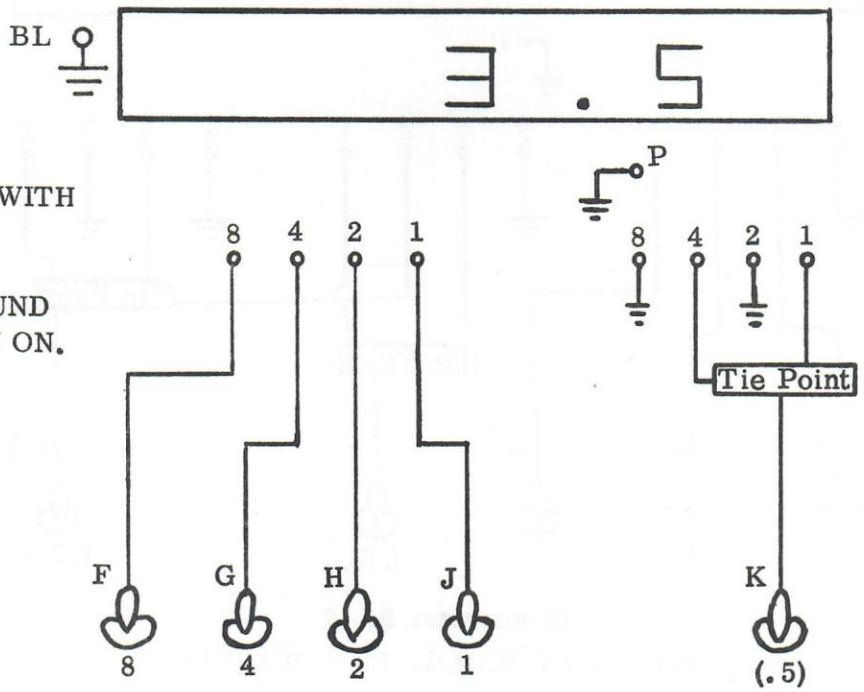
0.5 VOLT RESOLUTION

and also



0.25 VOLT RESOLUTION

The corresponding binary to BCD coding for use on the display is shown below:



NOTE: FOR UNITS WITH 12 KEY KEYBOARD The BL pin must be removed from GROUND to turn the DISPLAY ON.

Experiment 20 - 1

0.5 VOLT RESOLUTION DISPLAY

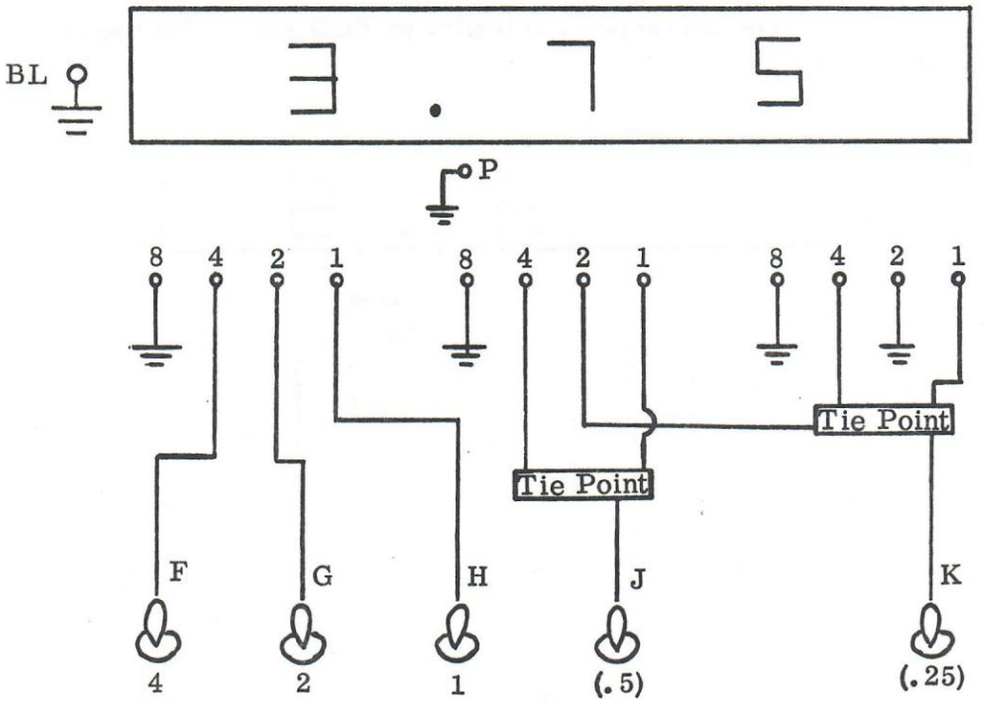
PROCEDURE:

Wire the above logic on the Ed - Lab 900 and check various settings from 0.0 to 9.5. Enter the test data below:

F 8	G 4	H 2	J 1	K .5	DISPLAY
0	0	1	1	1	3.5

Change the wiring for a 0.25 volt resolution as shown below:

NOTE: FOR UNITS WITH 12 KEY KEYBOARD
The BL pin must be removed from GROUND
to turn the DISPLAY ON.



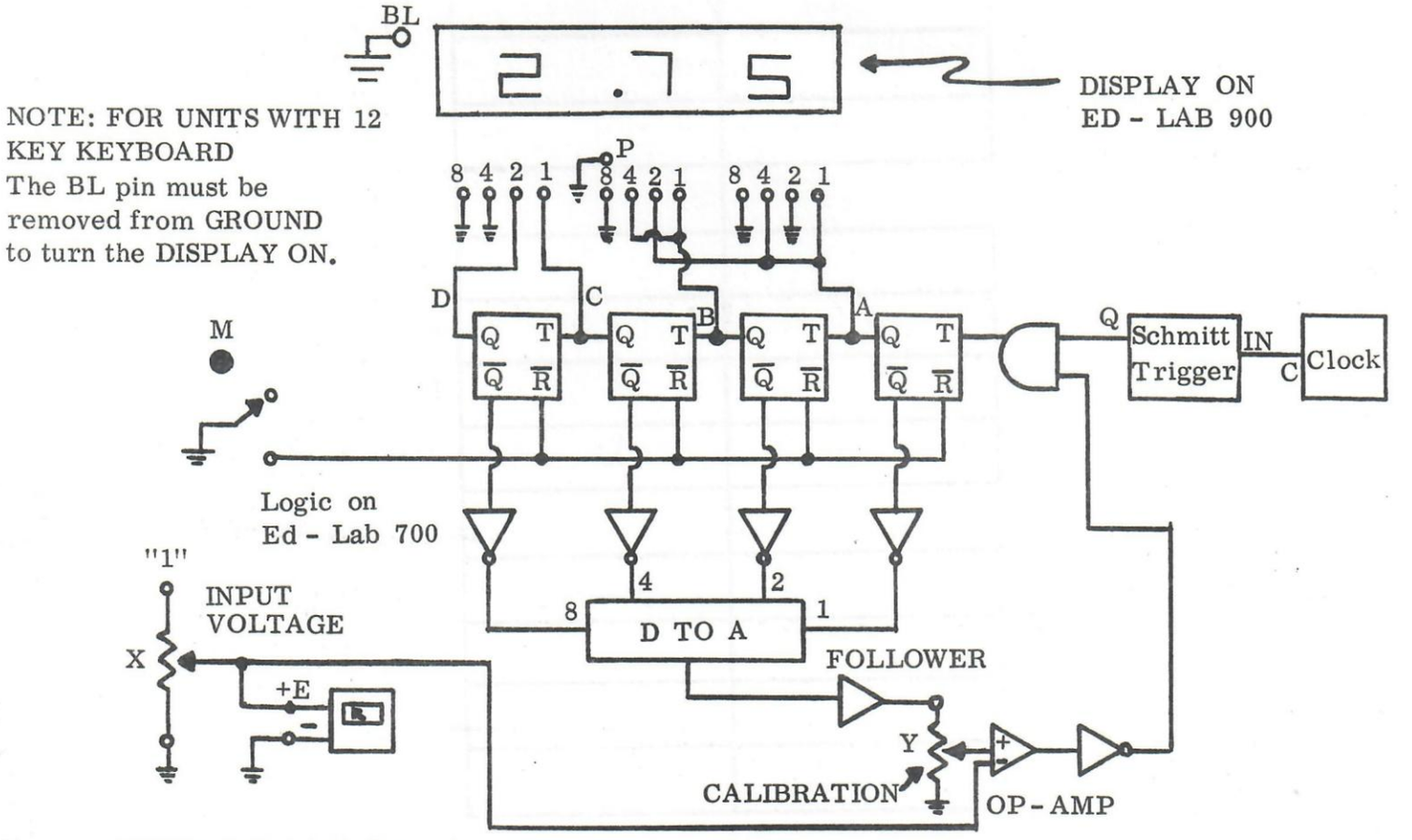
Experiment 20-2
0.25 VOLT RESOLUTION DISPLAY

Wire this logic and enter various test runs below:

F 4	G 2	H 1	J .5	K .25	DISPLAY
0	0	1	1	1	
0	1	0	1	0	
0	0	1	0	0	

Either of these display versions can be used in conjunction with the Digital Voltmeter Ed - Lab 700, Experiment No. 73.

Select the 0.25 volt resolution mode with the Ed - Lab 700 and the Ed - Lab 900 Display. The logic is shown below:



NOTE: 700 & 900 Grounds must be connected together.

Experiment No. 20 - 3

DIGITAL VOLTMETER WITH DISPLAY

PROCEDURE:

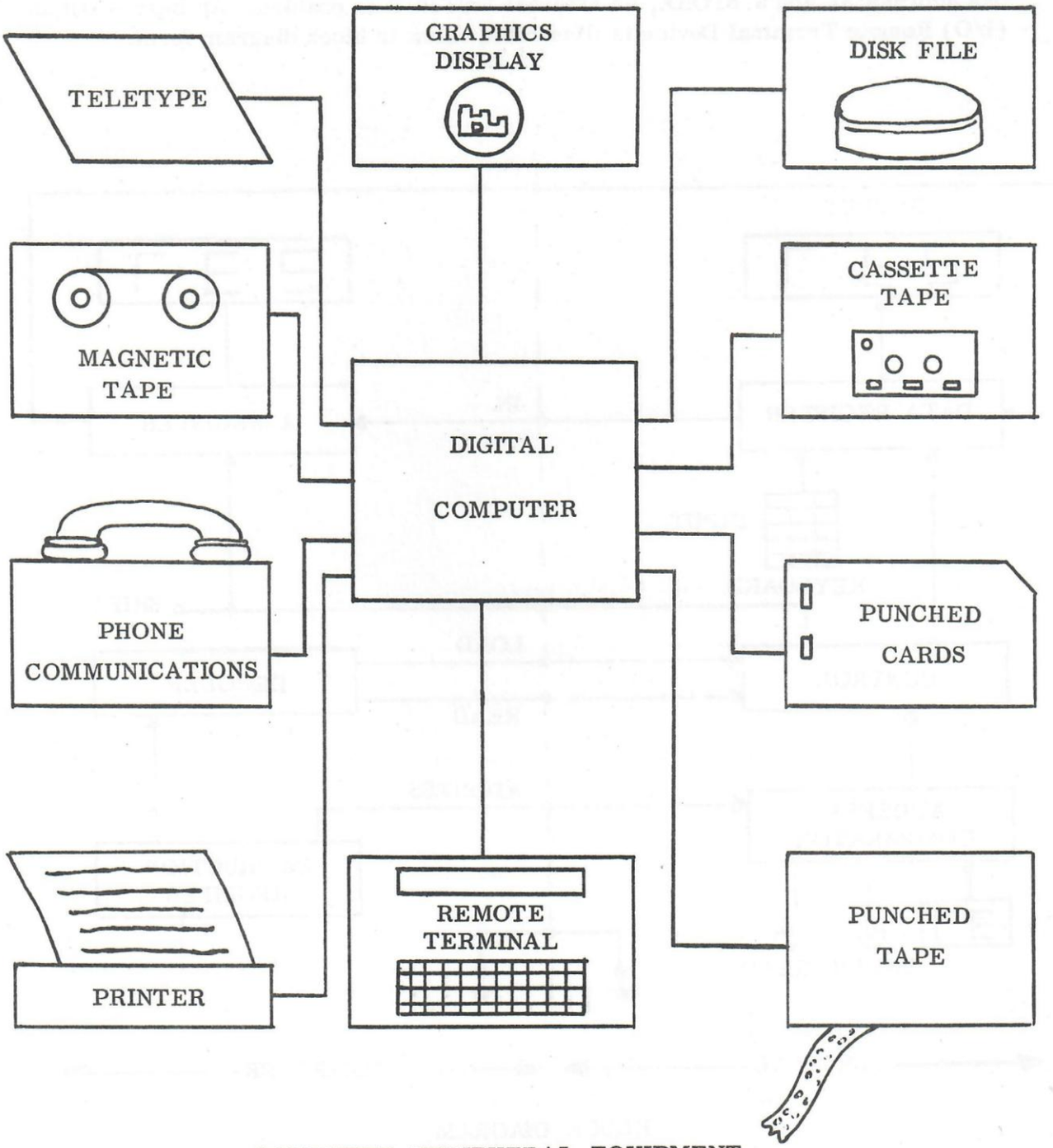
Wire the system and "Calibrate the Voltmeter" using Potentiometer Y. Set a voltage using Pot X. Reset using Switch M. Use a HIGH or LOW speed Clock.

Enter data below:

SET VOLTAGE	DISPLAY READING

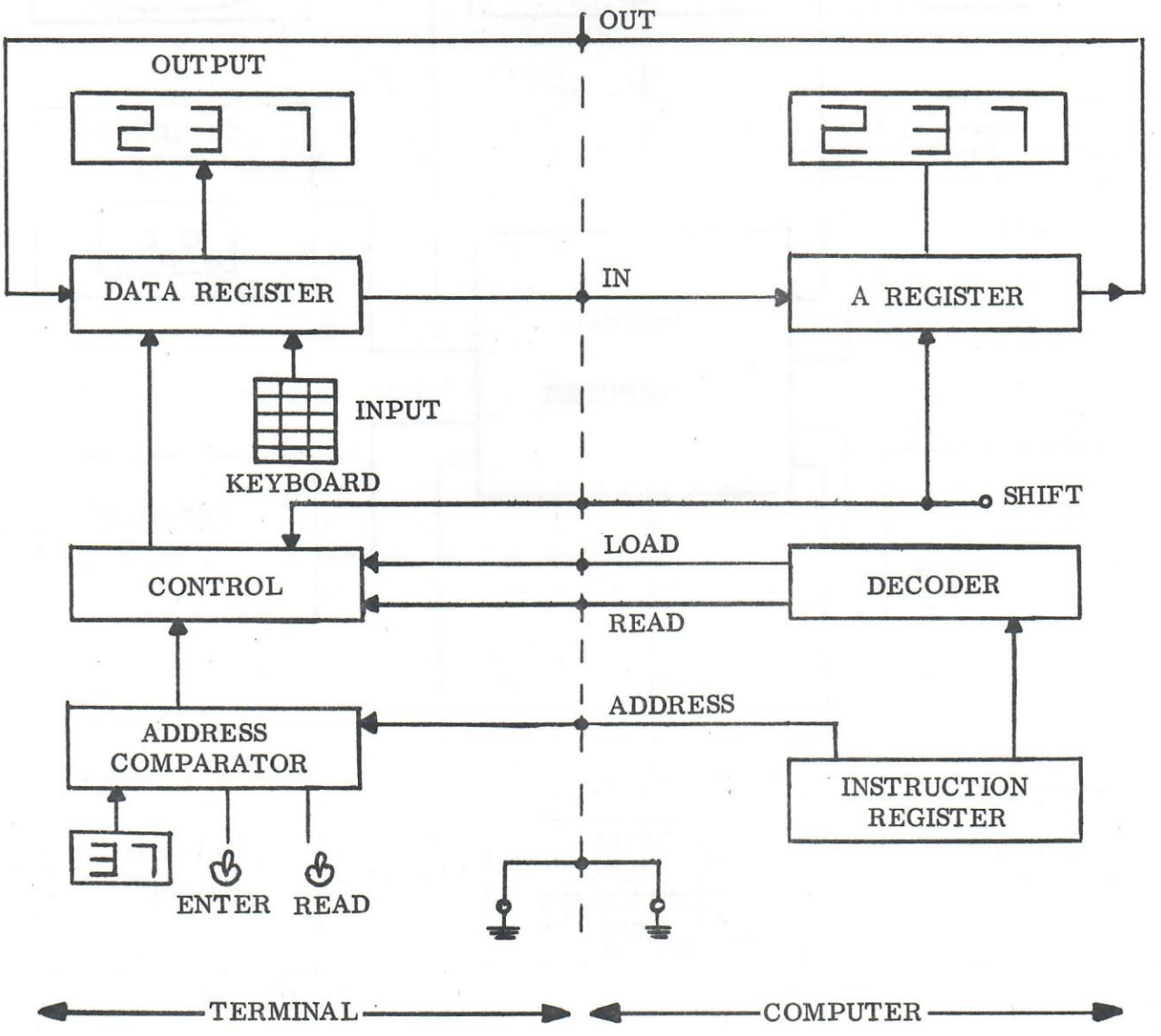
DESCRIPTION (REMOTE INPUT - OUTPUT TERMINAL):

The full capability of a Digital Computer is achieved only with the use of input and output "peripheral" equipment such as those indicated below:



COMPUTER PERIPHERAL EQUIPMENT

The Automatic Digital Computer (Experiments 8 and 9) transfers data between the Arithmetic Register and the Memory at the address in the Instruction. When using other external "peripheral" input - output devices such as a teletypewriter, punched paper tape device, magnetic tape, punched cards device, remote terminal, etc., a specific command or particular address can correspond to a specific external device. The system we will employ is to have address 37 correspond to the external device. When the instruction has address 37 with a LOAD instruction, an external INPUT is taken. When the instruction has address 37 and a STORE, an external OUTPUT is enabled. An Input - Output (I/O) Remote Terminal Device is illustrated below in block diagram form.



BLOCK DIAGRAM
REMOTE TERMINAL

The DATA REGISTER and DISPLAY are for INPUT and OUTPUT. The Data Register is set from the Keyboard (and could have been set from paper tape, magnetic tape, etc.). The Display is an Output (and could be a recording device, printer, punch, etc.). Two control switches are for ENTER (Input to Computer) and READ (Output from Computer). When the ENTER switch is UP, it indicates that the Data Register has valid data for the computer. An Address Compare determines an instruction address corresponding to this remote terminal (address 37). When there is an ENTER mode, a correct address, and a LOAD command, the data is transferred from the terminal Data Register to the Computer A Register as an INPUT.

When the READ Switch is UP, and there is a STORE command with Address 37, the data from the A Register is transferred to the terminal Data Register.

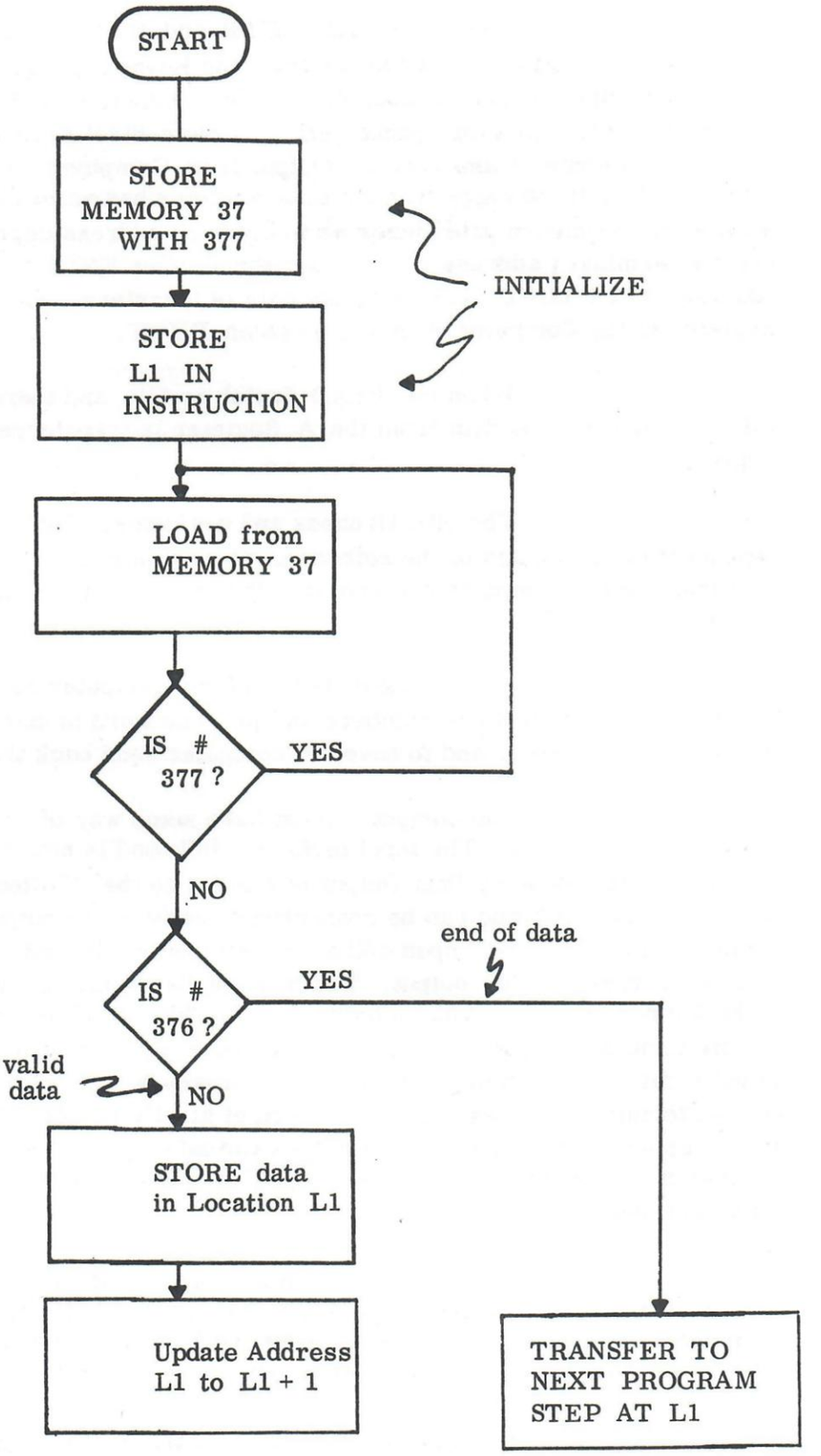
The effectiveness and usefulness of the remote terminal is dependent on the design of the software. Our main goal is to have the remote terminal use the computer and program it just as we might do at the computer directly.

A second goal is to use the computer as a "calculator", i.e., to send the computer some numbers and just one word or number for ADD, MULTIPLY, SQUARE-ROOT, etc., and to have the computer send back the answer.

The computer must have some way of "knowing" when valid input data is received. The input technique followed is now described. The electronic circuitry in the memory Data Output is similar to the "Collector Logic" elements in the Ed - Lab 700 and can be connected directly to the output of another logic element with (DTL) or "open collector" elements. In that case, either element logic at a "0" forces the "0" output. The Remote Terminal Output is connected directly to the Memory Output. The memory data in address 37 is set at all 1's so that the remote terminal output ("0" or "1") appears at the memory output. When there is no valid data, the terminal output is set at a steady "1". Thus, when reading the remote terminal address 37, a data word of all 1's or octal 377 specifies that no data is being read. Thus, 377 is not a valid data input. The computer program can be written to continuously read memory 37 until valid data is received and then to store that data.

The computer must also know when the "last" data is received. The convention that we have adopted is to transmit a 376 data word as the last word. Upon receiving this number, the computer will use the first data word received as the next program step and thus "transfer" to the remotely entered program.

The flow chart is given on the following page:



FLOW CHART

REMOTE TERMINAL PROGRAM

The writing into the remote terminal is done by writing into Memory Location 37 which will automatically write into the remote terminal unless the terminal register or shift pulse is inhibited at the terminal.

The logic for the remote terminal as a display or output only device is given in the logic on the following page.

The Memory Data Out is connected to the Instruction Register Data - In and the "Shift I Register Pulse" is also connected. The remote terminal I Register thus always presents the current instruction. The Decoder then detects a LOAD or STORE instruction. Logic AND gates decode the 37 address and OPERATION Mode and enable the Decoder. The STORE 37 instruction enables an AND gate to permit "Shift A Register Pulses" to shift the data into this remote display. The Memory Data - In is connected through the ADDER B Input to the A Register and the DISPLAY. Thus, any computer instruction to STORE 37 will transfer data to the REMOTE OUTPUT DISPLAY TERMINAL.

PROCEDURE (Remote Output Display Terminal):

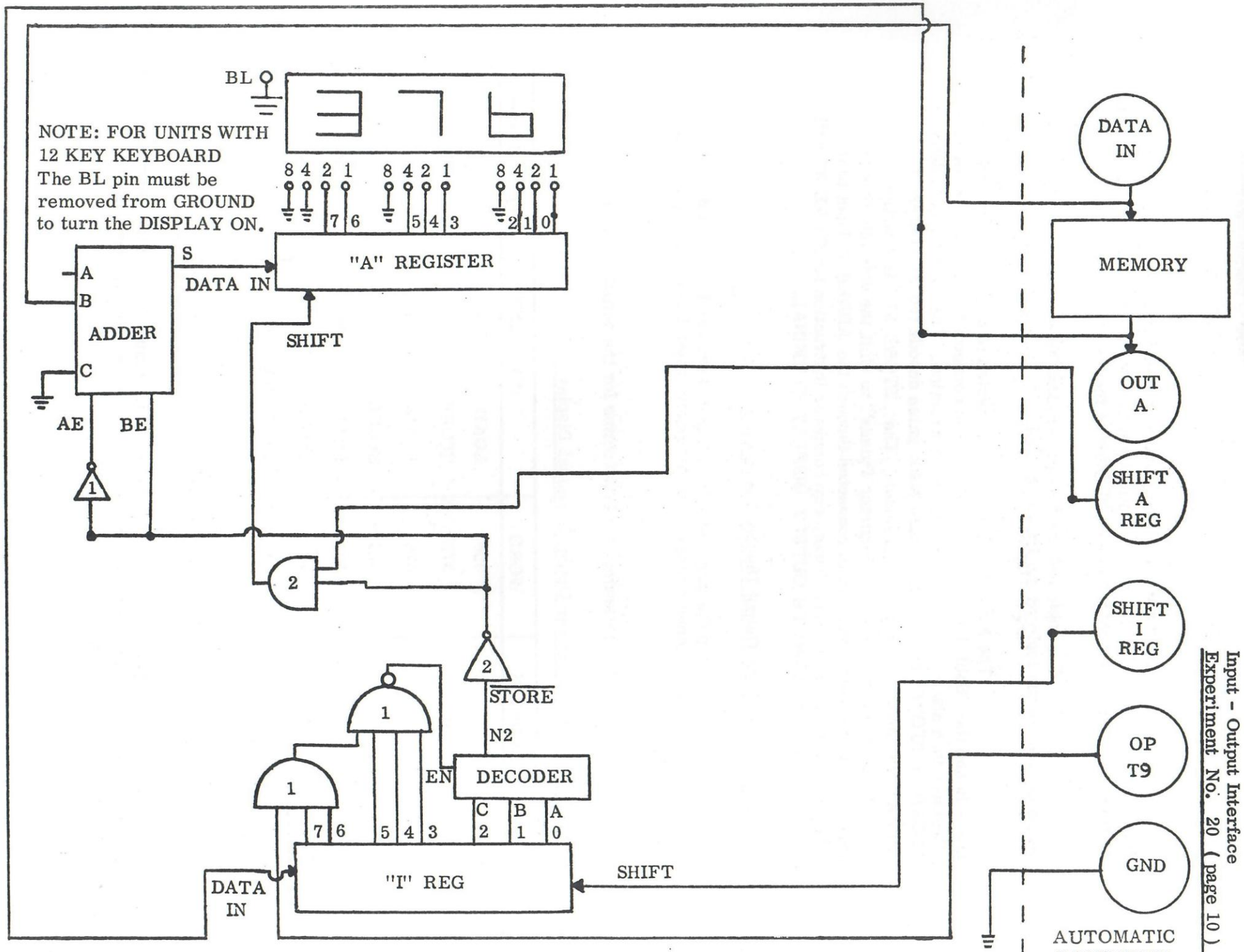
Wire the Logic shown in Experiment 20-4 on an Ed - Lab 700 and an Ed - Lab 900. Connect these to an Automatic Digital Computer (Ed - Lab 700 and 900).

The following is a test routine for the output terminal.

TEST PROGRAM: Remote Output Terminal Display

	ADDRESS	WORD	COMMENT		Remote Display
START	00	066	LOAD	303	000
	01	372	STORE		303
	02	076	LOAD	305	303
	03	372	STORE		305
	04	106	LOAD	323	305
STOP →	05	372	STORE		323
	06	303	DATA IN	(C)	
	07	305	ASCII	(E)	
	10	323	CODE	(S)	

Enter the above routine into the computer using the normal computer Keyboard method and run the test. Observe the display and enter the results in the table on page 11.



Experiment Logic 20-4 REMOTE DISPLAY

REMOTE OUTPUT DISPLAY:

NOTE: The computer is operated in the One Cycle Mode, observe the data appear on the display of the remote terminal as the computer is advanced through the program.

WORD	DATA

The Remote Display would normally present a long sequence of numbers. It is not desirable to repeat the LOAD and STORE instructions. Instead, an instruction loop is preferred. Consider the problem where fifteen Output Data words are stored in the Memory locations starting at 01 and it is desired to remotely display these words in sequence. The program is given on the following page.

EXERCISE:

1. Prepare the Flow Chart for this program after referring to the coded program.

Ed-Lab 900 : PROGRAM CODING FORM

PROGRAM : REMOTE TERMINAL OUTPUT ROUTINE

ADDRESS		WORD			COMMENTS
0	0	1	7	1	TRA (TO 20)
0	1				OUTPUT DATA
0	2				
0	3				
0	4				
0	5				
0	6				
0	7				
1	0				
1	1				
1	2				
1	3				
1	4				
1	5				
1	6				
1	7				
2	0	3	0	6	LOAD (INITIAL # OUTPUT DATA) } INITIALIZE
2	1	2	2	2	STORE }
2	2	ø	ø	ø	LOAD (DATA)
2	3	3	7	2	STORE (IN PERIPHERAL)
2	4	2	2	6	LOAD (ADDRESS DATA)
2	5	3	1	5	ADD (UP-DATE + 010)
2	6	2	2	2	STORE (NEW ADDRESS)
2	7	2	1	1	TRA
3	0	0	1	6	CONSTANT (INITIAL LOCATION)
3	1	0	1	0	CONSTANT
3	2				
3	3				
3	4				
3	5				
3	6				
3	7				NOT USED - PERIPHERAL ADDRESS

Enter this program with the following data. Record the output data.

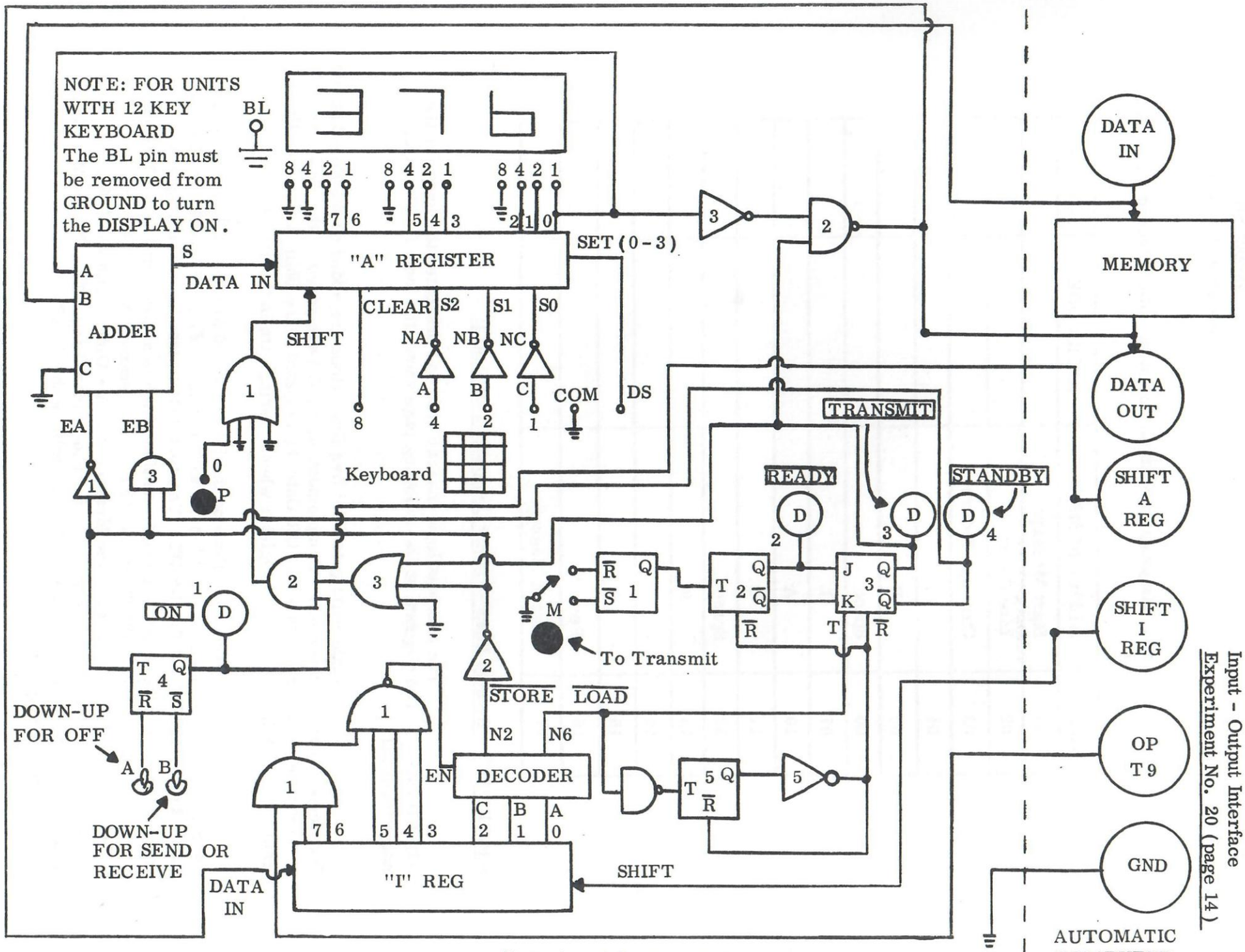
WORD	ALPHA NUMERIC	ASCII CODE	CHECK
01	Start Message		
02	Line Feed		
03	Carriage Return		
04	E		
05	D		
06	Space		
07	L		
10	A		
11	B		
12	Space		
13	9		
14	0		
15	0		
16	Period		
17	End of Message		

PROCEDURE (Remote Input - Output Computer Terminal):

The previous logic diagram (20-4) presented the logic for the output terminal. The diagram on the following page shows the logic for the input - output terminal.

Eight logic gates and five flip - flops are added on the Ed - Lab 700. The Keyboard and 3 Inverters (Complement A, B, C) are added on the Ed - Lab 900 to enter data in the usual manner. OR Gate #1 is added in the Shift A Register line to enable the P Switch to shift the register when entering data in the remote terminal.

Three flip - flops on the Ed - Lab 700 are used for start - stop control. Switch M drives Flip - Flop #1 which triggers Flip - Flop #2 (and Display Lamp 2) for READY TO TRANSMIT. The next LOAD decoder signal triggers Flip - Flop #3 (at the negative going transition which is at the start) to the TRANSMIT (Q = 1) state. The TRANSMIT signal (FF #3, Q) enables NAND #2 allowing the A Register bit 0 to be applied to the Memory DATA - OUT. With a FF #3, Q = 0, the NAND #2 output is always at a "1" for non-transmitting. This TRANSMIT signal is applied via the added OR #3 to AND #2 enabling the A REG Shift Pulse to be applied to the A Register.



Experiment Logic 20 - 5
REMOTE TERMINAL

At the end of the LOAD cycle, the transition via an Inverter (NAND #3) triggers a Flip - Flop #5 One - Shot which resets ($\overline{R}$) Flip - Flops #2 and #3.

When "transmitting" data as an INPUT to the computer, it is desired to retain the information in the A Register. This is done in the standard manner via the ADDER A input. The inverted STORE signal enables the B input for computer OUTPUT. When there is no STORE input, the A is enabled for LOAD (input to computer from the terminal).

Flip - Flop #4 controls the transmission of the A Reg Shift Pulses via AND #2. The shift pulses are enabled when FF #4, Q = 1 (Lamp 1 ON). Switch B - down sets the FF#4, Q = 1 for steady INPUT and OUTPUT. Switch A - down resets FF#4, to Q = 0 for a steady OFF.

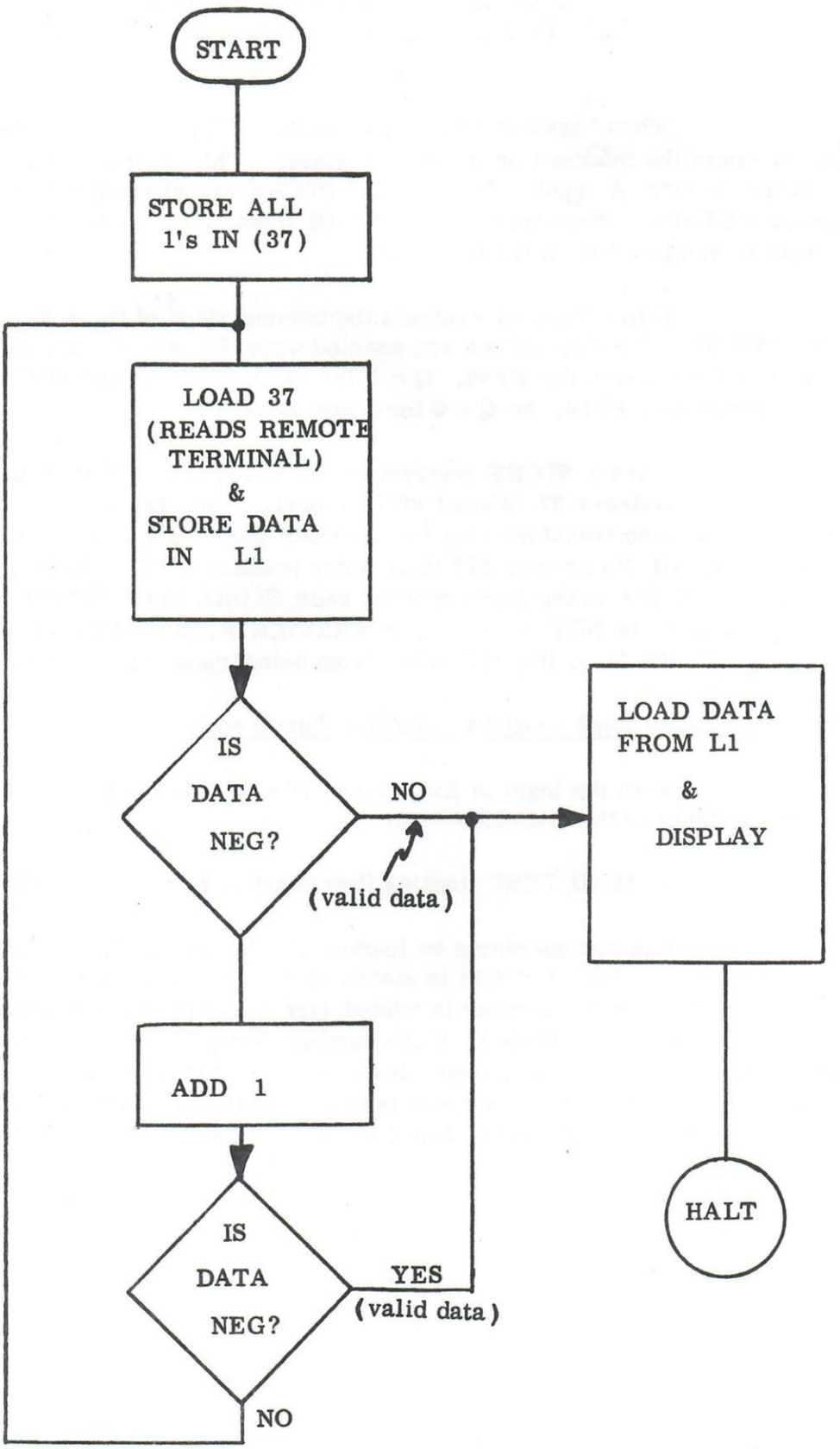
When a STORE instruction is used for an OUTPUT to the remote terminal, a STORE address 37 is used with the data. This transfers data to the remote terminal but also transfers data into memory position 37. Because it is required to have an all 1's or data 377 in memory position 37 for INPUT, it is necessary to STORE 377 in the memory after each STORE in the remote terminal. The STORE command via NOT #2 is used to TRIGGER FLIP - FLOP #4 to inhibit every alternate STORE (i.e. the 377 data) from being transferred to the terminal.

PROCEDURE (Remote Input - Output Computer Terminal):

Wire the logic in Experiment 20 - 5. Use the previous logic and add the logic elements in the diagram shown.

A READ TEST Routine flow chart is given on the following page.

The routine starts by loading all 1's into memory. The memory position 37 is read (LOAD) and data is stored in L1. If the number is positive, then it is valid "data" and the number is loaded (from L1) to the A Register and displayed and the operation is Halted. If the number is negative, it is either valid or not valid. A "1" (i.e. 001) is added. If the number is then positive, it must have been a 377 or invalid. If the number is negative, it was valid data. On invalid data, the computer returns to read (LOAD 37) the remote terminal again.



FLOW CHART: REMOTE TERMINAL - READ TEST

The coded program is given below:

PROGRAM: REMOTE TERMINAL READ - TEST ROUTINE

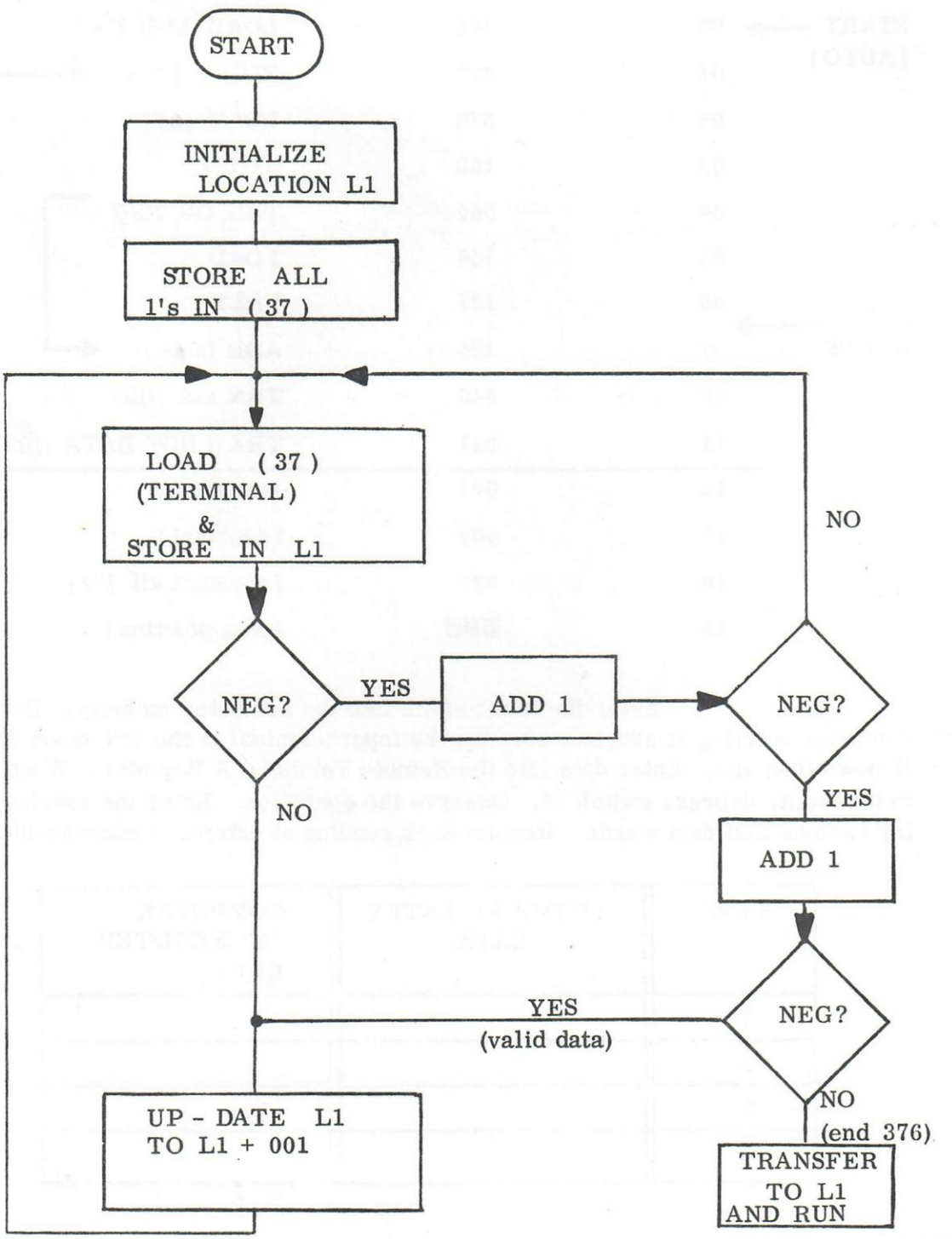
	ADDRESS	WORD	COMMENTS
START → (AUTO)	00	146	LOAD ALL 1's
	01	372	STORE (37)
	02	376	LOAD (37)
	03	152	STORE
	04	060	TRN ON NEG
	05	156	LOAD
	06	137	HALT
AUTO HALTS →	07	125	ADD 001
	10	040	TRN ON NEG
	11	011	TRA (NOT DATA)(02)
	12	001	(constant)
	13	000	(constant)
	14	377	(constant all 1's)
	15	ØØØ	(data position)

Enter the test routine into the computer memory. Run the computer starting at address 00. Set the input terminal in the ON mode (Switch B down then up). Enter data into the Remote Terminal A Register. When desired to transmit, depress switch M. Observe the operation. Enter the results below for various test data words. Restart each reading at computer address 00.

TEST	TERMINAL ENTRY DATA	COMPUTER "A" REGISTER DATA
1		
2		
3		
4		

PROCEDURE (Remote Programming):

A complete remote terminal operation program will read many input data words, perform a series of instructions, and transfer answers to the output terminal. The basic "interactive terminal" flow chart is given below:



REMOTE INTERACTIVE TERMINAL OPERATING PROGRAM

The routine is similar to the test routine. However, after data is stored, the address is "up-dated" for the next address location. A second addition and Transfer - On - Negative is used to detect a 376 data input. Upon detection of 376, the program is started at location L1.

The Coded program is given on the following page. Location 25 to 36 are reserved for the external program. Thus ten program entries are possible.

Enter the Interactive Terminal Program (Program given on next page) into the basic computer and start the computer operation at 00.

Now using only the remote terminal, enter programs and observe the results. The following example is given to solve for X = B + 2A.

REMOTE PROGRAM FOR X = B + 2A

ADDRESS	WORD	COMMENTS
25	326	LOAD (32)(B)
26	335	ADD (33)(A)
27	335	ADD (33)(A)
30	372	STORE (37) B + 2A (ANSWER to output terminal)
31	371	TRA to 00 (for new readings)
32	002	B (example)
33	007	A (example)
34	376	(exit word)
35		
36		

The above words are entered in sequence and the computer automatically stores the data starting in location 25. When word 376 is entered, the computer starts at program step 25 and computes B + 2A (002 + 007 + 007 = 020) and transfers the output (answer) to the remote terminal in step 30. It then proceeds to step 00 of the main program and continues to search for new remote data for another problem.

Ed-Lab 900 : PROGRAM CODING FORM

PROGRAM : INTERACTIVE TERMINAL PROGRAM					
ADDRESS		WORD			COMMENTS
0	0	2	2	6	LOAD L1 } INITIALIZE L1
0	1	0	5	2	STORE }
0	2	2	4	6	LOAD # 377 } INITIALIZE memory
0	3	3	7	2	STORE (37) } to all 1's
0	4	3	7	6	LOAD 37 (READ REMOTE)
0	5	2	5	2	STORE
0	6	1	2	0	TRANS ON NEG (TON)
0	7	0	5	6	LOAD (POSITIVE - VALID)
1	0	2	3	5	ADD (L1 + 1)
1	1	0	5	2	STORE (IN INSTRUCTION)
1	2	0	3	1	TRA (04)
1	3	2	1	5	ADD (001)
1	4	1	5	0	TON
1	5	0	3	1	TRA
1	6	2	1	5	ADD (001)
1	7	0	6	0	TON
2	0	2	4	1	TRA (START PROGRAM)
2	1	0	0	1	CONSTANT (001)
2	2	2	5	2	CONSTANT (L1)
2	3	0	1	0	CONSTANT
2	4	3	7	7	
2	5				}
2	6				}
2	7				}
3	0				} EXTERNAL PROGRAM
3	1				}
3	2				}
3	3				}
3	4				}
3	5				}
3	6				}
3	7				remote terminal address

Enter the remote program using the terminal. Observe the result.

A	B	X

Enter the same program with another A and B. Enter the results below:

TEST	A	B	Computer Answer B + 2A
1			
2			
3			

EXERCISE:

2. Prepare another program for remote entry. Enter the program and record the resulting data.

DESCRIPTION (Calculator Program):

The above program required a complete written program by the terminal operator. A "calculator operation" requires only the entry of data and one word to denote ADD, SUBTRACT, MULTIPLY, DIVIDE, SQUARE-ROOT, etc.

Consider the problem of such a program where the operator enters only two numbers. The first is the data and the second is the operation to be performed with the data on an internal computer register.

An example is given using the following codes:

CODES	
ADD	010
SUBTRACT	020
MULTIPLY	030
DIVIDE	040
SQUARE - ROOT	050
CLEAR	060

To obtain a solution to:

$$X = \sqrt{017 + 012}$$ (Note: Number in Octal)

we would enter the following:

STEP	1st (DATA)	2nd (CODE)	DISPLAY (ANSWER)
1	000	060	000
2	017	010	017
3	012	010	031
4	000	050	005

The operation is the same as with a calculator. Each intermediate result is displayed.

EXERCISE:

3. Prepare a Flow - Chart for the above calculator problem.

4. Prepare a Flow - Chart for a calculator which accepts 2 data inputs and an operation code where the operations are as given in the table on the following page.

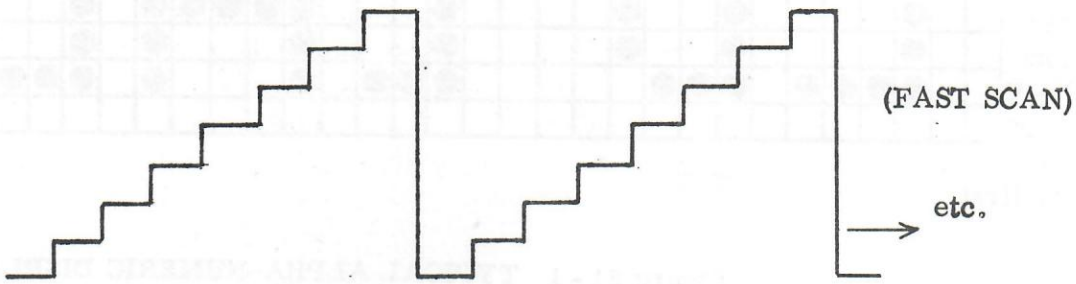
CALCULATOR OPERATIONS
A + B
A - B
B - A
2A + B
A + 2B
2A + 2B

OBJECTIVE:

You will wire a bit-by-bit memory output system to display an 8x32 DOT PATTERN on an OSCILLOSCOPE. You will enter 1 and 0 bits into the memory to display a sequence of letters or words.

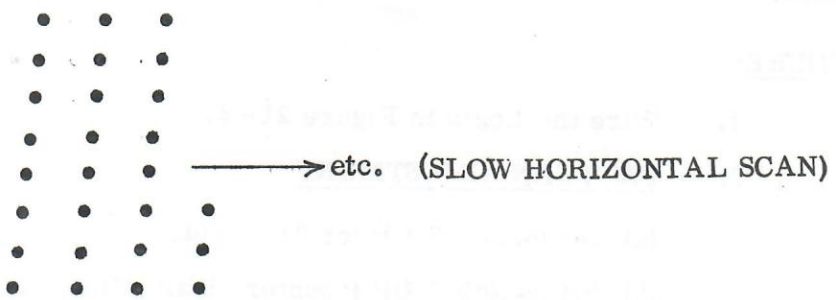
DESCRIPTION:

The Standard 3-bit Counter into a Digital to Analog Converter generates an 8-step function as would be seen on the Oscilloscope as shown below:



3 BIT D to A

As the horizontal sweep speed is decreased, this would appear as a sequence of 8 vertical dots as shown below:



DOT PATTERN

If some dots are blanked, then a recognizable pattern of letters or words will result. The Memory has a capacity of 256 bits which results in a display field of 8 dots high and 32 dots wide. Each Dot corresponds to an addressed bit. A "1" will turn the dot on and a "0" will turn the dot off or "blank" the dot for display.

The Dot Pattern is shown below:

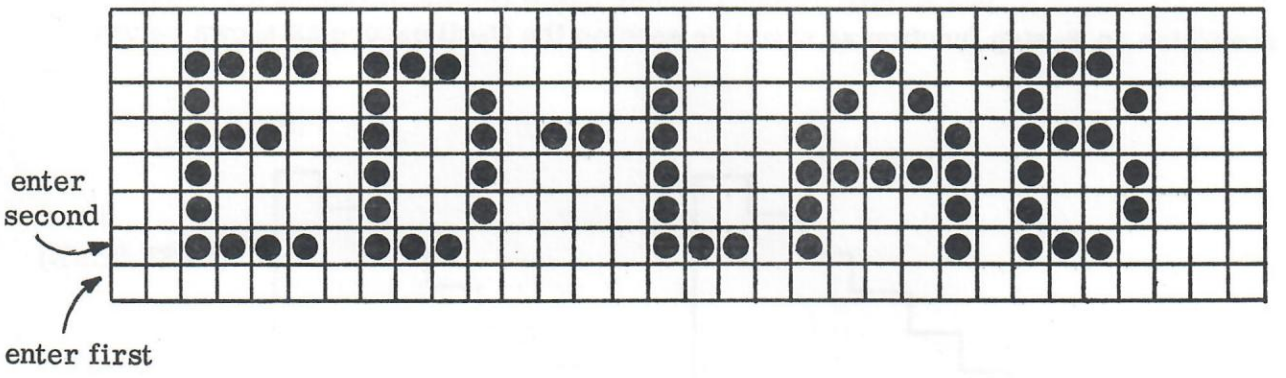


Figure 21-1 TYPICAL ALPHA-NUMERIC DISPLAY

The logic diagram for this experiment is shown on the following page in Figure 21-2. The Counter Addresses the Memory. Inverters #1, #2 and #3 and Flip-Flops #1 through #7 merely act as a high speed clock and could be replaced by a Pulse Generator. Counter Bits 0, 1, 2 drive the D to A Converter. The Full Adder is used as a selection logic gate which permits manual entry via Switch P through the "A" input when Switch H is DOWN; and automatic run clock pulses through the "B" input when Switch H is UP. The memory output data is Inverted (#4) and Solid-State Switch T1 on the Ed-Lab 700 holds the Analog Voltage output to "0 VOLTS" for a "0" memory bit, resulting in this dot being "moved down" and blanked from the pattern.

PROCEDURE:

1. Wire the Logic in Figure 21-2.
2. PATTERN ADJUSTMENT
   - (a) Set Switch F UP for "1" input.
   - (b) Set Switch J UP (counter clear off).
   - (c) Set Switch H UP for RUN.
   - (d) Press Switch P, and then release.
   - (e) All 1's are now entered in Memory.
   - (f) Adjust the Oscilloscope for a dot pattern.

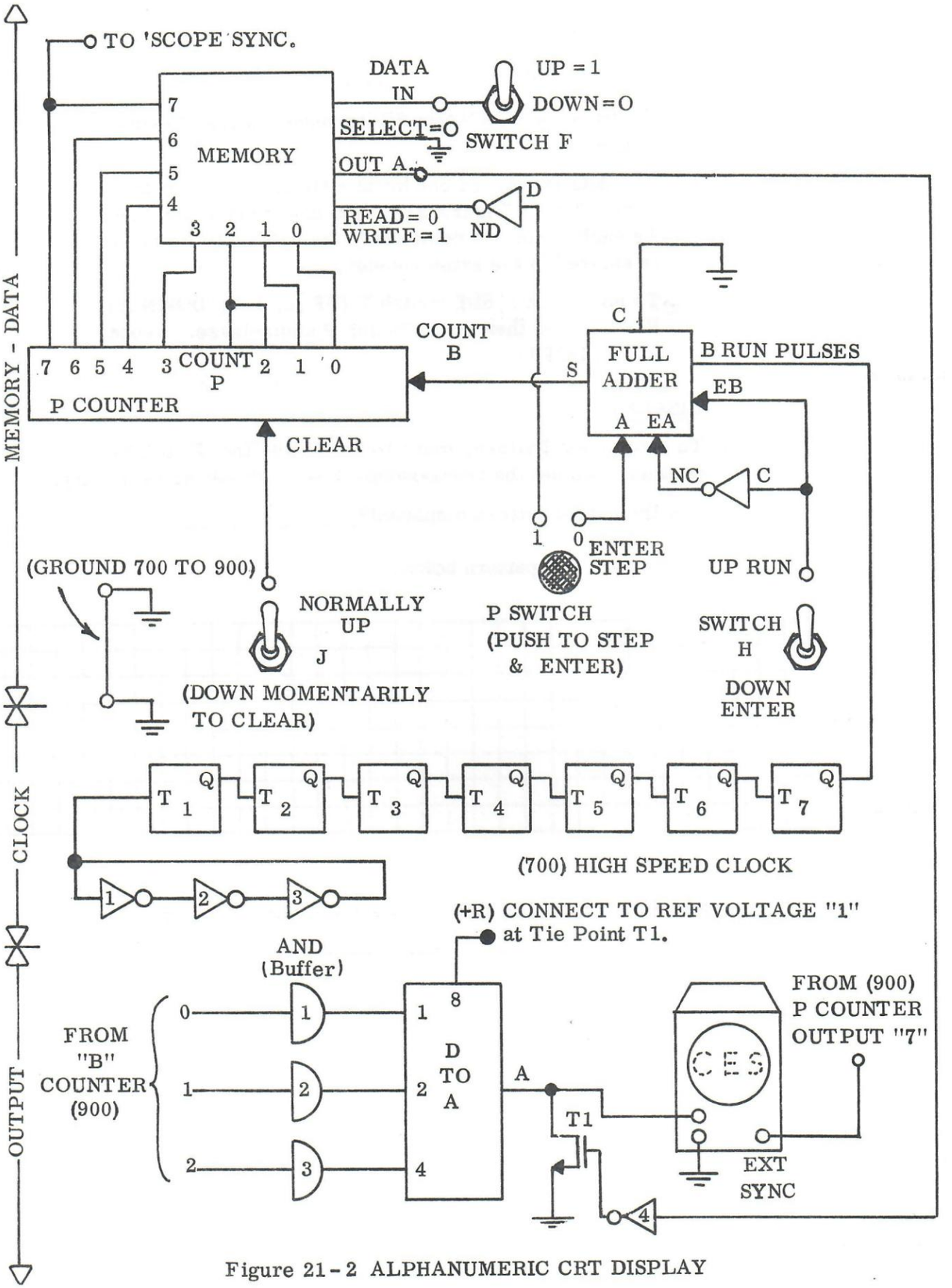


Figure 21-2 ALPHANUMERIC CRT DISPLAY

3. ENTER DATA

(a) Set Swtich H DOWN for Data Entry.

(b) Set Switch J DOWN then UP to reset Address Counter to zero.

(c) The data is entered one bit at a time, starting with the bottom left on Figure 21 - 1 and moving up the column. As each column is completed, the column to its right is entered in the same manner.

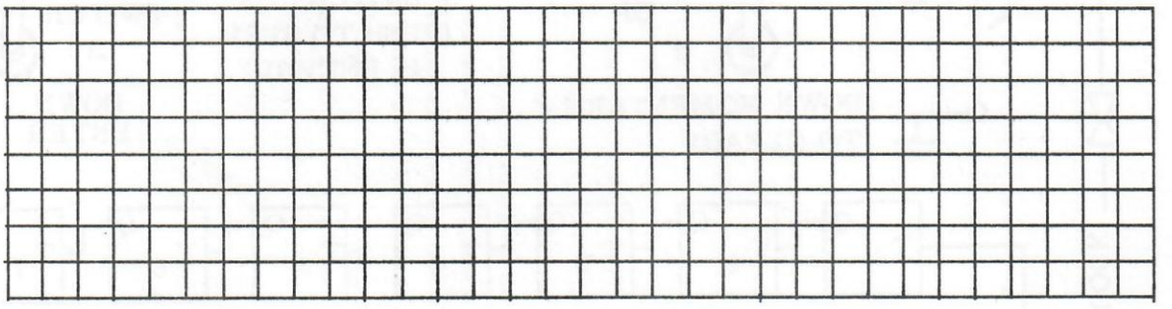
To enter a bit; SET Switch F (UP for LIT, DOWN for BLANK) and then press Switch P and release. Repeat for all DOTS.

4. DISPLAY

To display the Pattern, move Switch H into the UP (RUN) position. Adjust the Oscilloscope Sync or Sweep as necessary.

Are the proper letters displayed?_____

5. Set up your own pattern below:



6. Enter the DATA and RUN the display. Are the proper Letters displayed?_____

OBJECTIVE:

You will modify the alphanumeric format to display a square graphics or picture format and then you will enter and display picture data.

DESCRIPTION:

A previous experiment on Numerical Control consisted of controlling a device or generating a graphics display or drawing on an oscilloscope by moving a dot with Horizontal and Vertical Voltages. Another manner for generating a graphics display on an oscilloscope is similar to that used in the previous experiment. A square dot pattern is displayed on which any graph or picture can be shown. The diagram below shows a typical format:

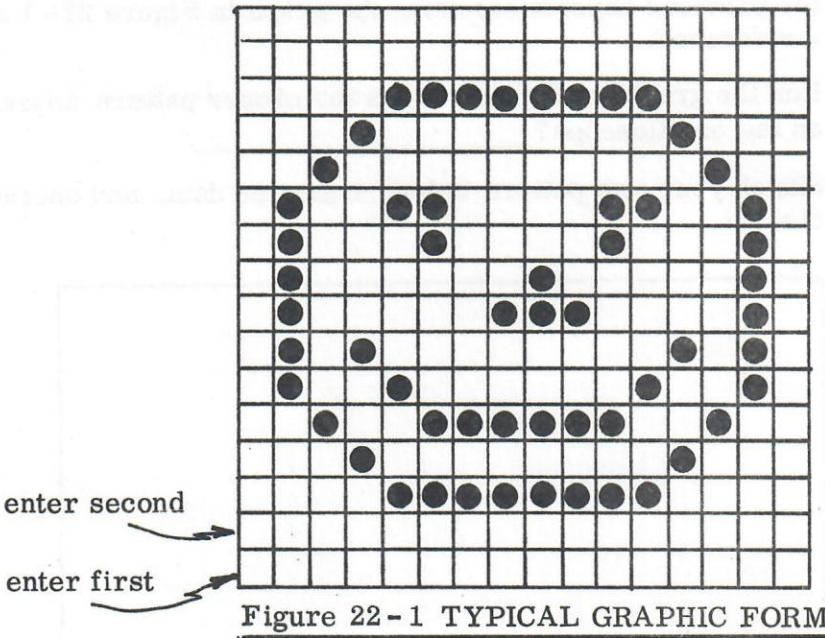


Figure 22-1 TYPICAL GRAPHIC FORMAT

The logic is shown below in Figure 22-2.

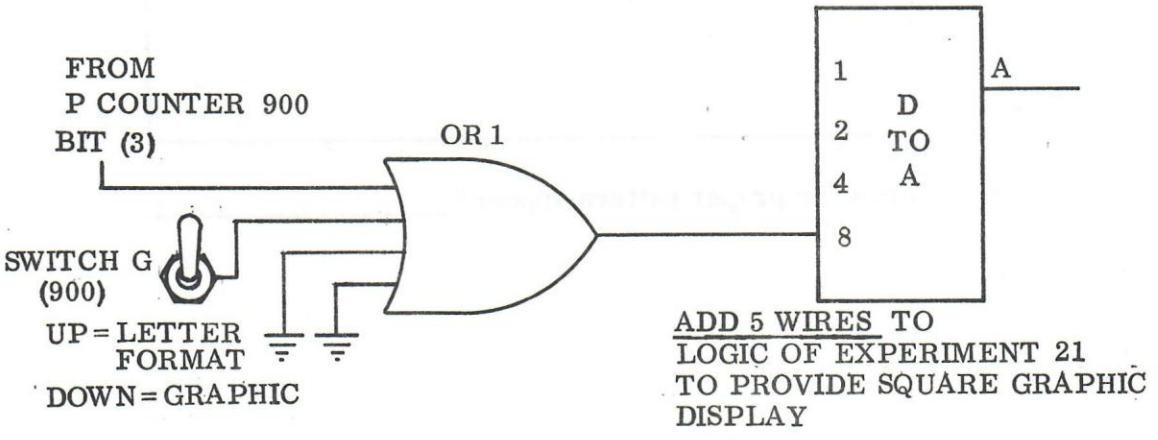
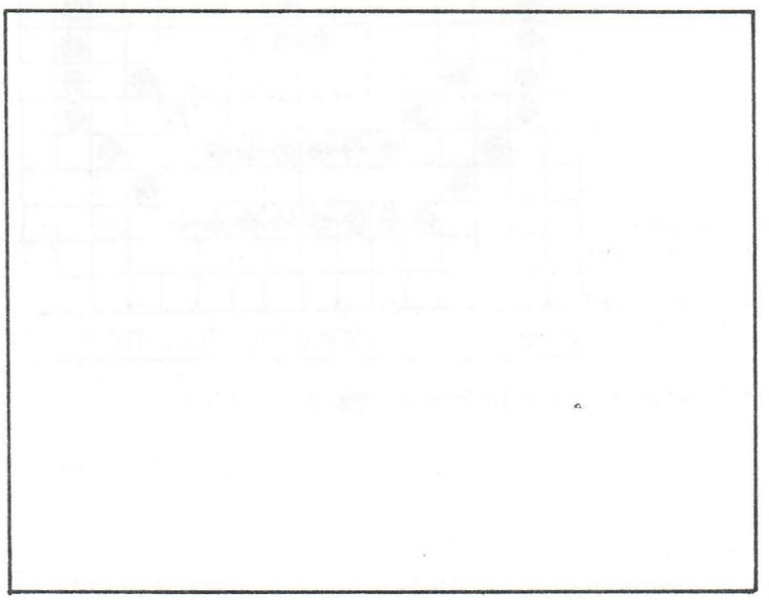


Figure 22-2 GRAPHIC FORMAT - ADDED LOGIC

The entire previous diagram is used with only one change and some additions. The D to A "8" bit is now used to increase the pattern height to 16 dots. The "3-bit" from the P-COUNTER is connected to the D to A "8" INPUT. The logic diagram shown uses an OR gate #1 so that Switch G can select a LETTER pattern or the square GRAPHICS pattern.

PROCEDURE:

1. Add the 5 wires in Figure 22 - 2 to the earlier diagram from Figure 21 - 2 for the Letter Format.
2. Enter the Graphics Data using the same procedures as in the previous experiment using the graph in Figure 22 - 1 as a reference.
3. Run the graphics display. Does the proper pattern appear on the oscilloscope?_____
4. Sketch your own pattern below, enter the data, and operate the display.
5. 
6. Does the proper pattern appear?_____

OBJECTIVE:

You will enter data in the memory corresponding to 256 time slots with 1 and 0 bits specifying ON and OFF tones entered in a MORSE CODE PATTERN. You will listen to the output over a loudspeaker.

DESCRIPTION:

The exact same memory logic used in the two previous experiments provides a sequential series of 1's and 0's which can be used for many applications. The pattern shown below is for a sequential series of bits used as "time-slots". A DOT is one bit or one space or time slot. A DASH is 3 spaces. One space is used between symbols (DOT or DASH) and 3 spaces are used between letters. A 256 time slot pattern is used as shown in the figure below:

CQ CQ DE WA2CDS WA2CDS

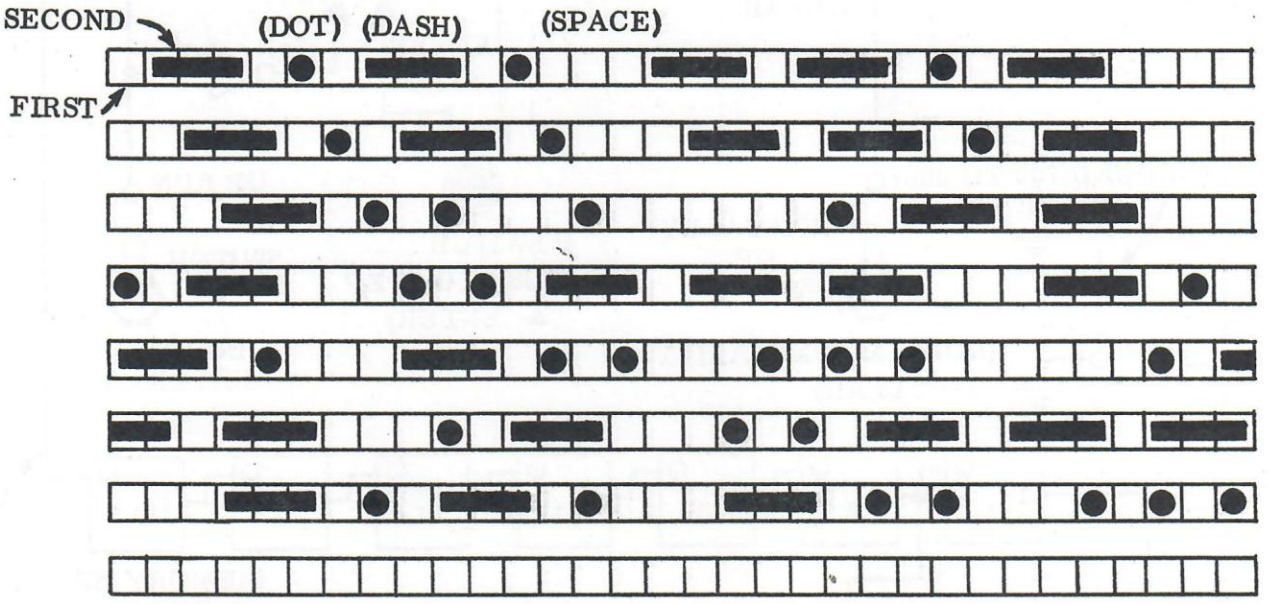
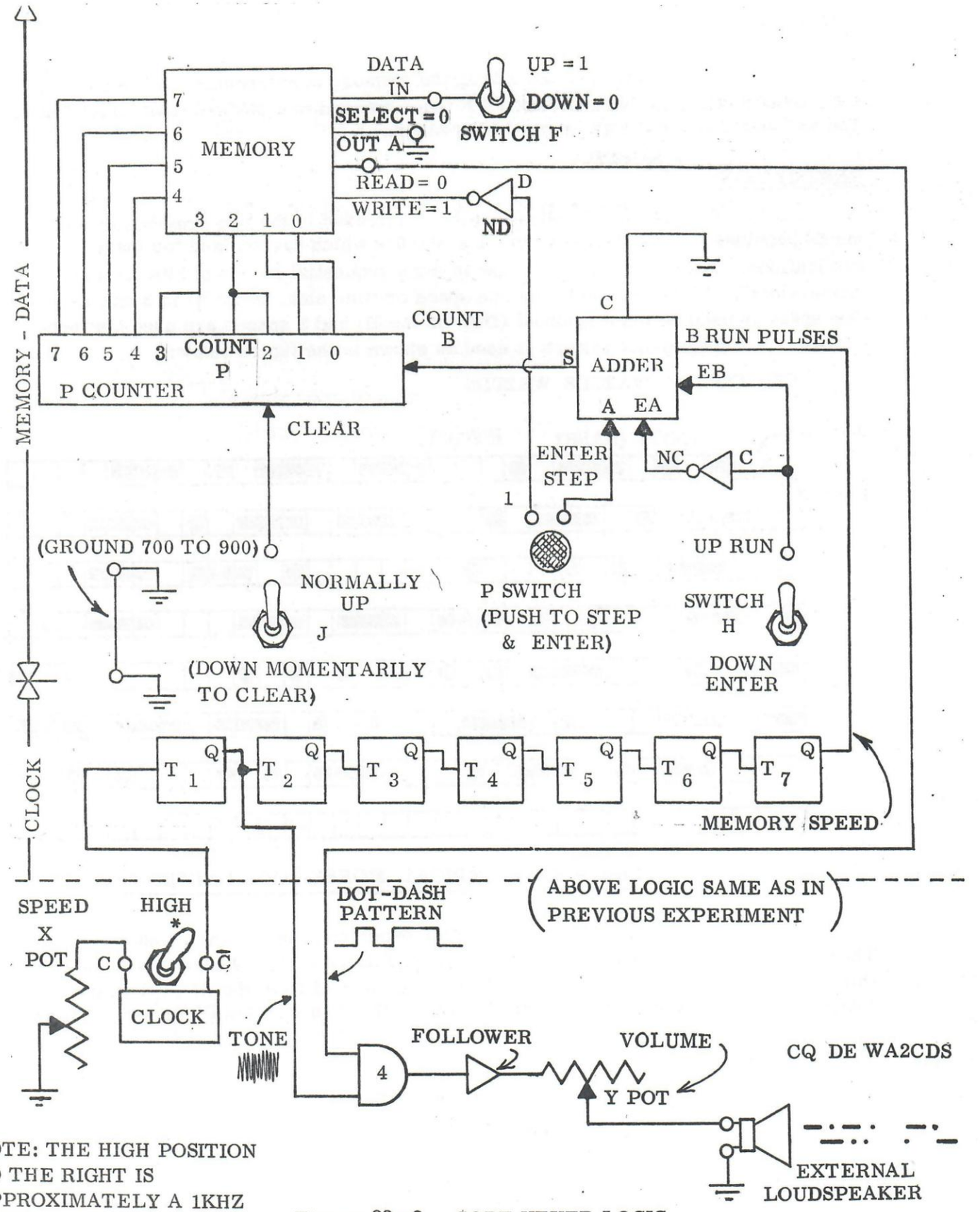


Figure 23 - 1 TYPICAL MORSE CODE PATTERN

The logic diagram for this experiment is shown in Figure 23 - 2. The same basic memory logic that was employed in the CRT Displays is used in this system. The basic clock is now slower and derived from the CLOCK on the Ed-Lab 700. The AND #4 is used to "Modulate" the tone with the DOT-DASH Pattern.

PROCEDURE:

1. Wire the Logic in Figure 23 - 2.
   Use an external Loudspeaker.



NOTE: THE HIGH POSITION TO THE RIGHT IS APPROXIMATELY A 1KHZ CLOCK. LATER VERSIONS HAVE A CENTER POSITION AT 10KHz.

Figure 23 - 2 CODE KEYER LOGIC

2. Enter Data as before:

   Set Switch H DOWN for manual entry.

   Set Switch J DOWN then UP for address reset.

   Enter Bits Using Switch F and Push Button Switch P.

3. When all data has been entered:

   Set Switch H UP for Run

   Adjust the X POT for Speed and Frequency.

   Adjust the Y POT for volume.

4. Does the audio output correspond to the Morse Code pattern recorded in memory?_____

5. The MORSE CODE is presented below for reference. Devise your own message and enter the code in the memory for automatic "Keying".

A · —	N — ·	1 · — — — —
B — · · ·	O — — —	2 · · — — —
C — · — ·	P · — — ·	3 · · · — —
D — · ·	Q — — · —	4 · · · · —
E ·	R · — ·	5 · · · · ·
F · · — ·	S · · ·	6 — · · · ·
G — — ·	T —	7 — — · · ·
H · · · ·	U · · —	8 — — — · ·
I · ·	V · · · —	9 — — — — ·
J · — — —	W · — —	0 — — — — —
K — · —	X — · · —	
L · — · ·	Y — · — —	
M — —	Z — — · ·	

OBJECTIVE:

You will connect a logic system to the Keyboard and REGISTER "A" Display System to provide automatic entry and shifting of data when a Key is depressed.

DESCRIPTION:

Earlier experiments such as the Automatic Computer provided for Keyboard entry of octal data into the A Register where the operator was required to depress the P Switch five times to shift the data five bits to the right (end around) which resulted in an effective shift of three bits to the left. The logic shown on the following page in Figure 24 - 1 in this experiment automatically generates the 5 SHIFT pulses when a Key on the Keyboard is depressed. After the 5 SHIFT pulses are generated, a SET (0-2) pulse is generated and data is entered.

The "0" Key is actually coded as a TEN or binary (1010). When the "0" Key is depressed, it is desired to shift 5 times and then enter a zero. Thus, it is necessary to eliminate the "2" output bit or the "S1" INPUT on the A REGISTER when a 1010 code is received. AND #1 removes the "2" output (S1) when an 8 ("0" when active) is present. The CLEAR signal for the A-Register is decoded by NOT #4 and NAND #2 for the MINUS (-) KEY which is coded twelve or 1100. Flip-Flops #5 and #6 generate an ON signal when the delayed strobe DS is obtained. Flip-Flops #2, #3 and #4 provide the MODULO 5 COUNT which then triggers a "one-shot" using Flip-Flop #7 and NOT #1 to generate a short pulse to "turn-off" the ON signal and to generate the SET (0-2) pulse through NOTS #2 and #3.

PROCEDURE:

1. Wire the logic shown in Figure 24 - 1.
2. Set-in various numbers and observe the operation. Enter MOST SIGNIFICANT OCTAL DIGIT FIRST.

ENTER DATA	CHECK
1 2 3	
2 4 7	
3 0 1	
0 2 4	
0 0 6	

3. Does the system operate properly? _____

NOTE: FOR UNITS WITH 12 KEY KEYBOARD
The BL pin must be removed from GROUND
to turn the DISPLAY ON.

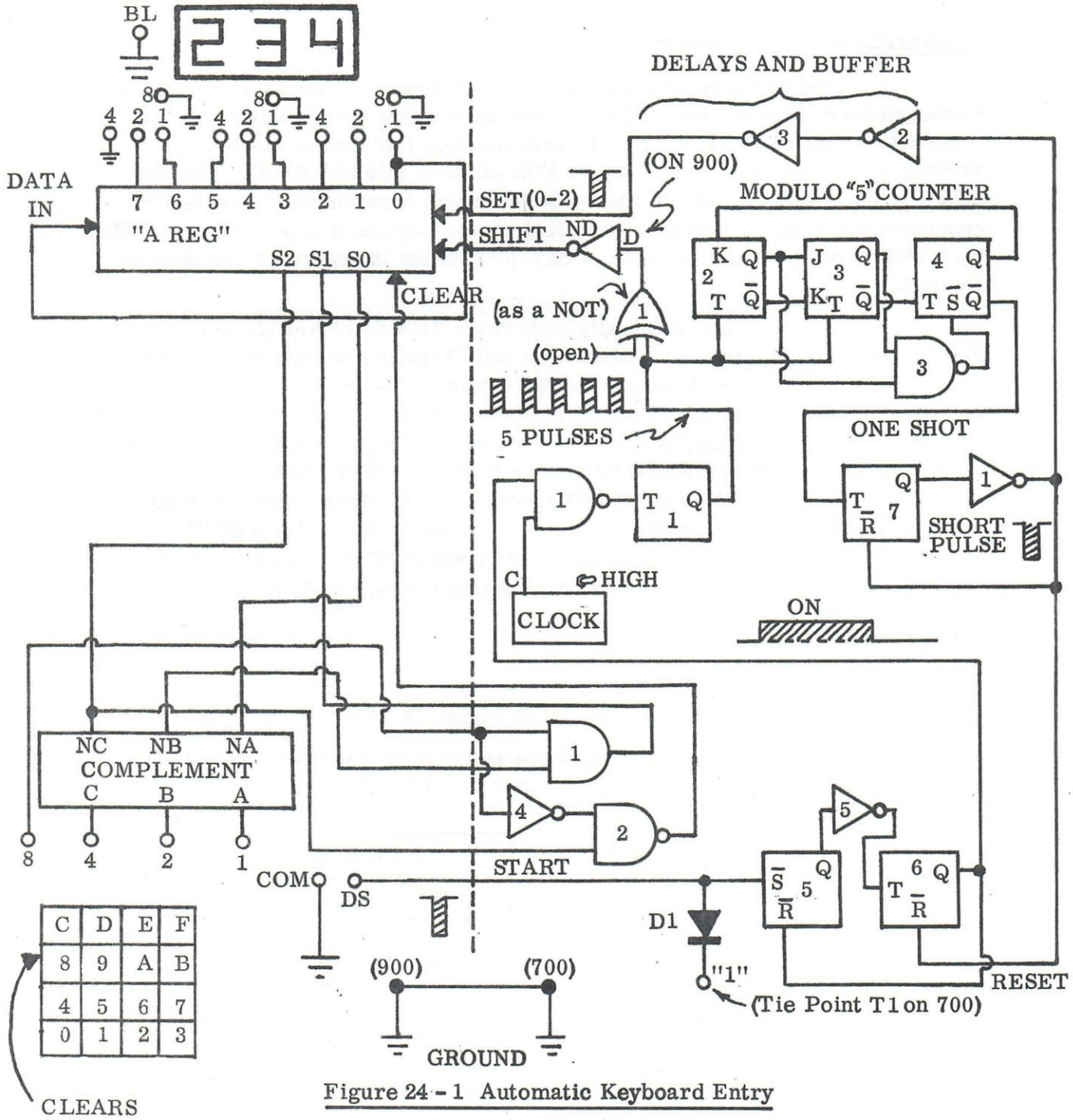


Figure 24 - 1 Automatic Keyboard Entry

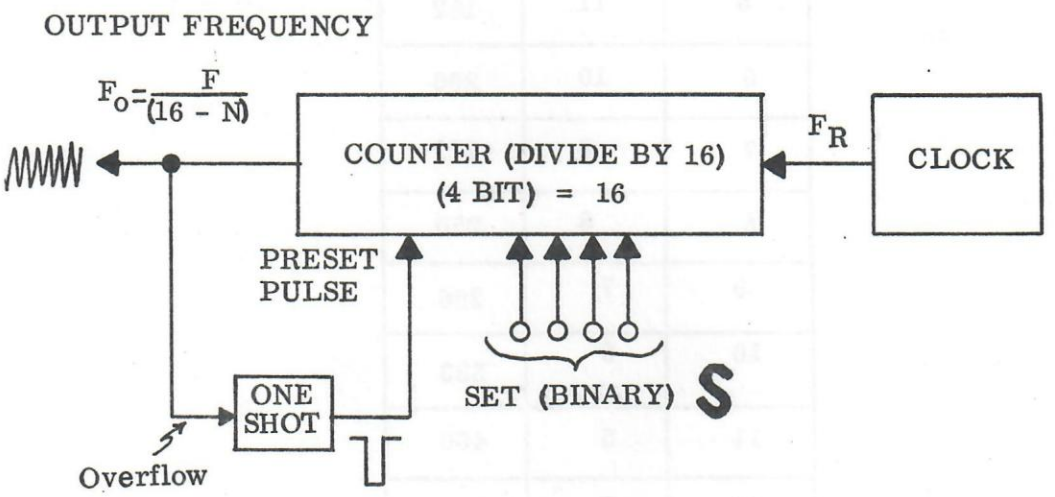
OBJECTIVE:

You will review the concept of the DIGITAL FREQUENCY SYNTHESIZER and test a Frequency Synthesizer and a Phase - Locked - Loop.

DESCRIPTION:

Variable Frequency Clocks are usually designed where the frequency is changed by changing a resistor (using a Potentiometer), or by changing a capacitor (Variable Capacitance), or by changing a Voltage (resulting in a variable charging current). A Digital Frequency Synthesizer uses a single fixed frequency clock, usually crystal controlled for high accuracy, with a digital counter presetting or decoding system. The resultant frequencies are as accurate as the crystal controlled clock source.

The basic system is shown in the Block Diagram below and referencing the Table on the following page.



BLOCK - DIAGRAM

DIGITAL FREQUENCY SYNTHESIZER

A Clock has a reference frequency F_R which is applied to the input of the Counter (4 bits in the illustration). The 4-bit counter normally divides the input frequency by 16. If the output "change from 1 to 0" or "overflow" is used to SET the counter to a number S then the frequency division is by (16 - S). The table shows the division-number and the resultant output frequency.

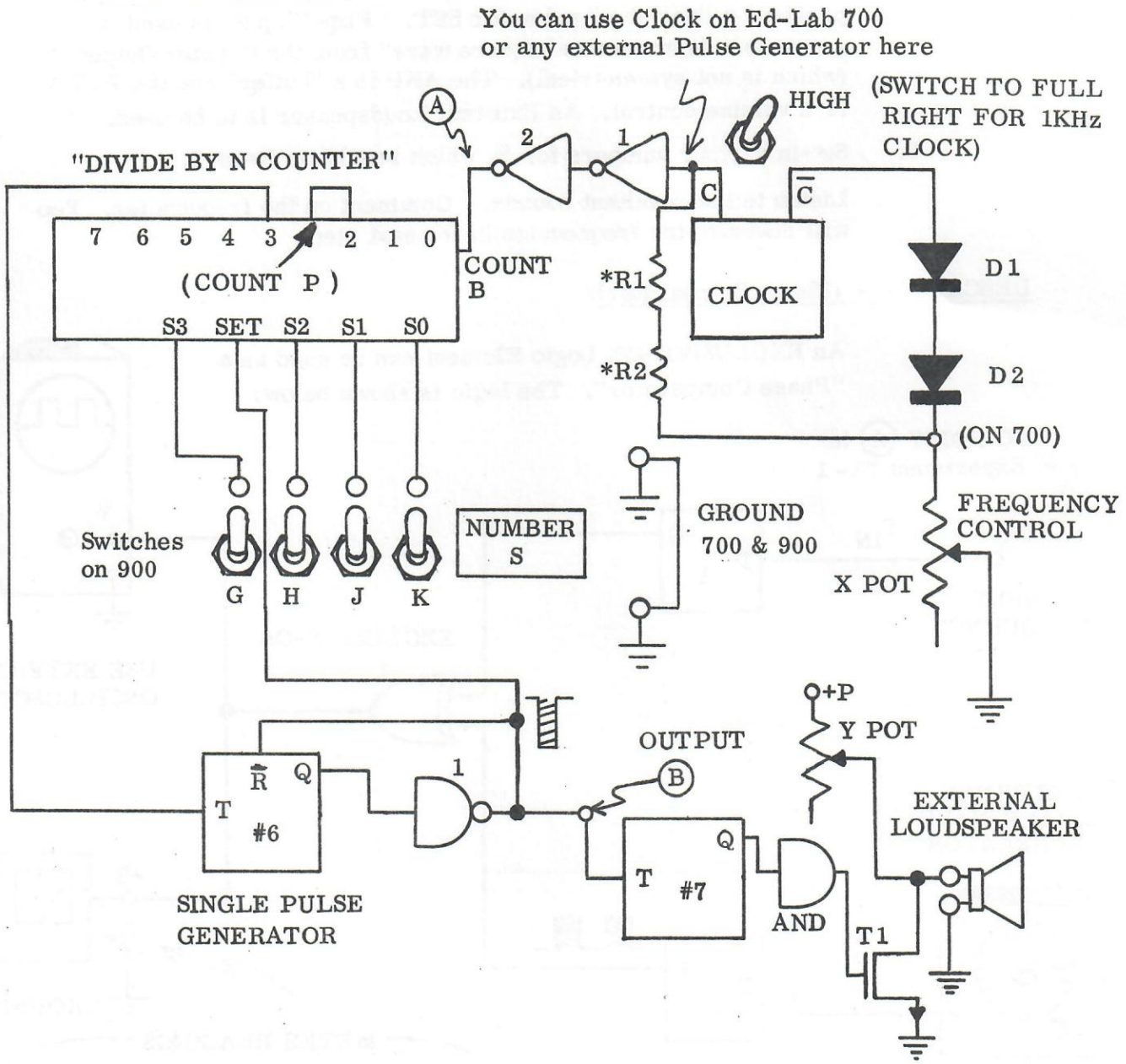
TABLE: OUTPUT FREQUENCIES

S =	÷ by N (16-S)	F_o
0	16	125
1	15	133
2	14	143
3	13	154
4	12	167
5	11	182
6	10	200
7	9	222
8	8	250
9	7	286
10	6	333
11	5	400
12	4	500
13	3	667
14	2	1000
15	1	2000

For F_R = 2000 HERTZ

$$F_o = \frac{F_R}{N}$$

*NOTE: Add R1 and R2 if 2000 Hz output frequency cannot be achieved. (See Text)



Experiment 25 - 1 Digital Frequency Synthesizer Logic

PROCEDURE (FREQUENCY SYNTHESIZER):

1. Wire the Logic in the figure for Experiment 25-1.

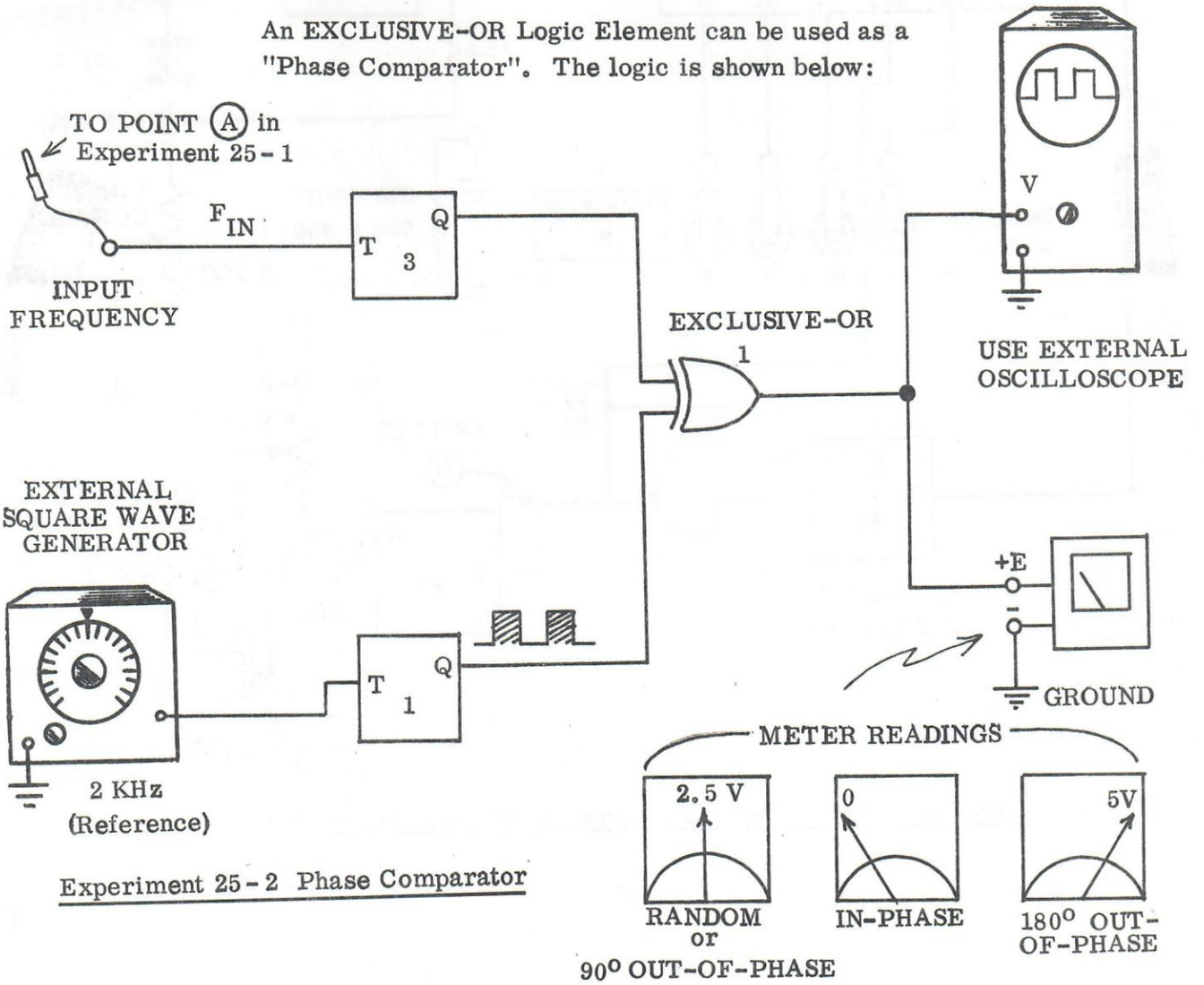
   Note that the Switches G, H, J, K "set-in" the binary number, S. Pot X controls the reference frequency. Flip-Flop #6 with NAND #1 provides a "one shot" pulse for SET. Flip-Flop #7 is used to generate a "symmetrical square wave" from the Counter Output (which is not symmetrical). The AND is a "buffer" and the POT Y is a volume control. An External Loudspeaker is to be used.

2. Set-in various numbers for S, which results in N=16-S.

   Listen to the resultant sounds. Comment on the frequencies. You will measure the frequencies in the next step.

DESCRIPTION (Phase Comparator):

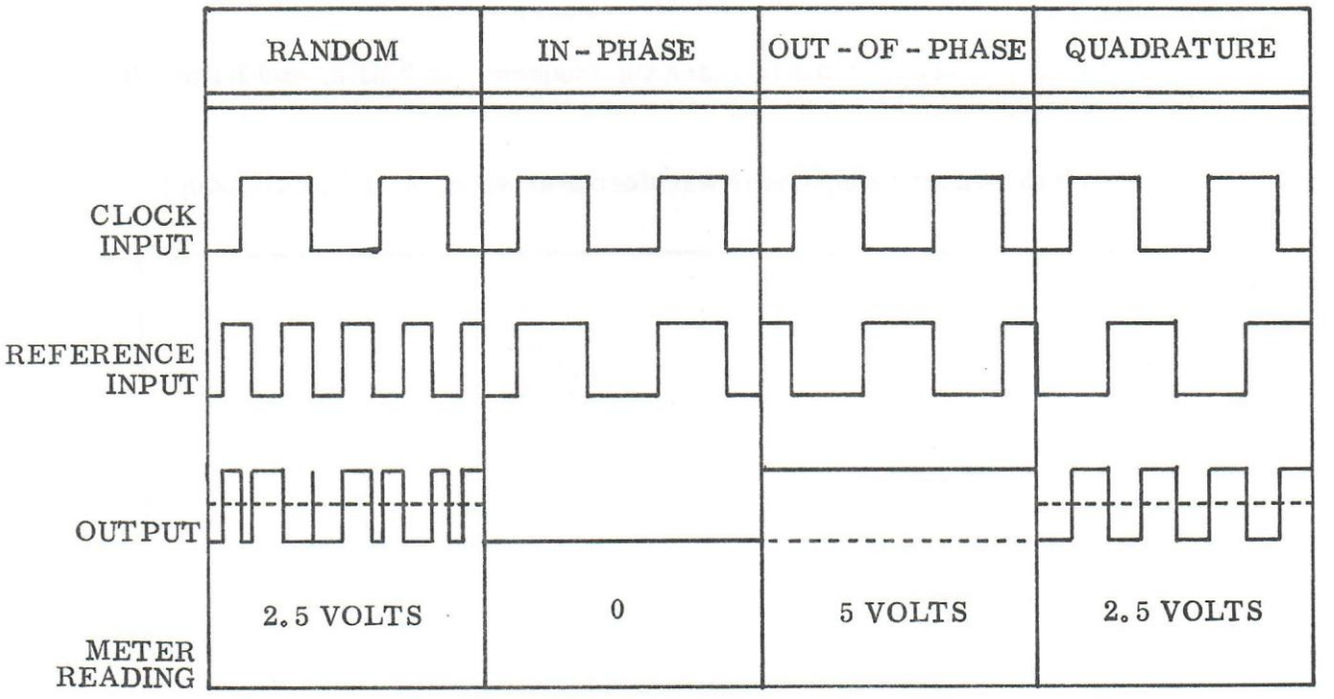
An EXCLUSIVE-OR Logic Element can be used as a "Phase Comparator". The logic is shown below:



Experiment 25-2 Phase Comparator

When both the measured INPUT FREQUENCY and the REFERENCE Generator Frequencies are Random or non-related, the EXCLUSIVE-OR output is either at a "1" or a "0" an equal number of times, and the meter output (which reads the average voltage) will read the average voltage between 0 and 5 volts and will read 2.5 volts.

The two flip-flops are used to provide a symmetrical square wave for both the input frequency and the reference clock. When both frequencies are exactly equal and "in-phase", the output of the Exclusive-OR will always be "0" and the meter will read zero volts as indicated in the diagram below. When both frequencies are the same with "opposite phases", the EX-OR will be a "1" at all times and the meter will read 5 volts. When the two frequencies are identical but "90 degress out of phase" or in "quadrature", the meter will read 2.5 volts. See the diagrams below:



When the meter reads a steady 2.5 volts, the two frequencies are not related. When the meter slowly moves from 0 to 2.5 and 5 and back to 0, the frequencies are the same with slowly shifting "Phases". By reading the frequency setting on the External Generator, your will then be able to check or measure the frequency being measured at f_{IN}. Using this Phase Comparator above, however, it is not possible to tell the difference between a reference oscillator frequency that is too high or too low.

PROCEDURE (PHASE DETECTOR):

1. Do not remove the wiring in Experiment 25 - 1. Add the wiring for the Phase Comparator in Experiment 25 - 2. Use an external Signal Generator and an external Oscilloscope.

2. Connect the INPUT FREQUENCY "lead" to the Inverter #2 Point (A) at the clock output. Set the EXTERNAL Square Wave Generator at 2KHz. Adjust the Signal Generator Pulse Amplitude for good triggering. You should use the oscilloscope to monitor the waveform output Q of Flip-Flop #1 while adjusting the amplitude.

3. Adjust the Ed-Lab Clock using the X POT until a Phase Comparison is indicated by a 0 volt, or 5 volt, or slowly varying 0 to 5 volt reading. At this point, the clock frequency is adjusted to 2 KHz. Note that the two Inverters #1 and #2 are buffers to prevent any frequency change if "loading" the Clock at output C by Phase Comparator, Oscilloscope, or Flip-Flop.

   Can you properly adjust the clock? _____

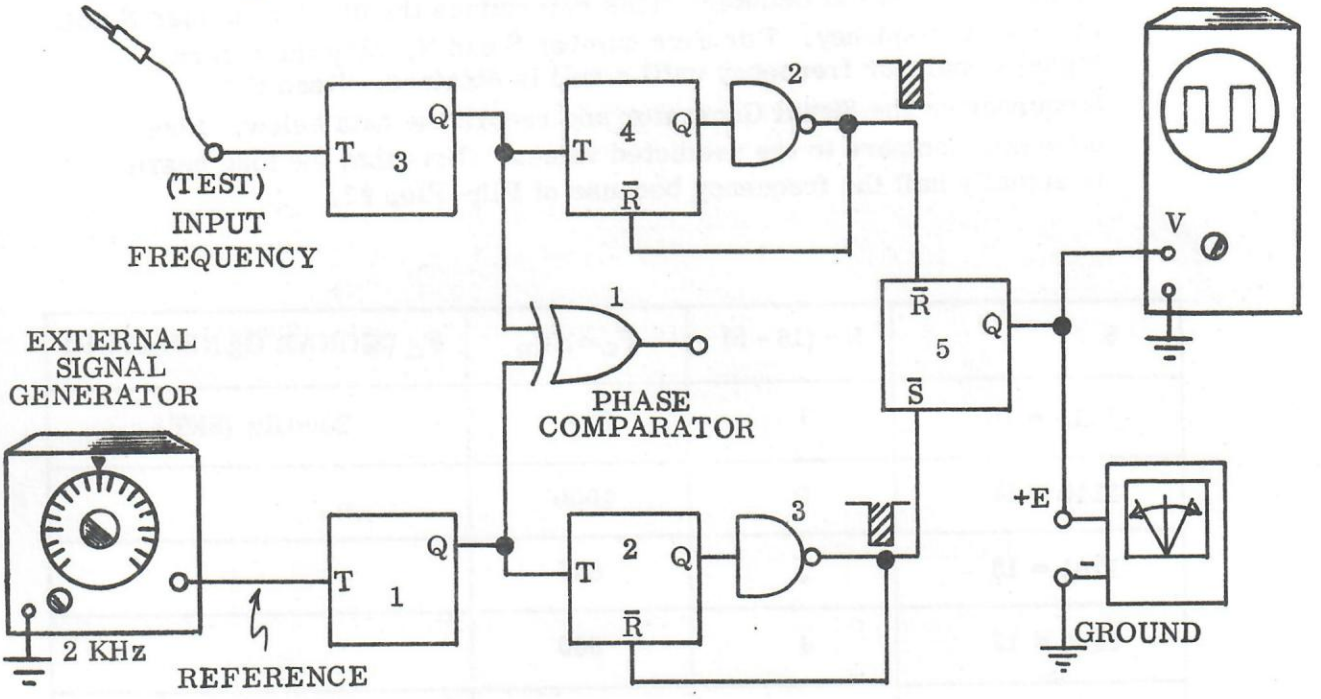
   Note: If you are not able to reach the frequency of 2 KHz, add R1 and R2 shown in Figure 25-1.

   Sketch the Phase Comparator waveforms as seen on the oscilloscope.

DESCRIPTION (Frequency Comparator)

The disadvantage of the phase comparator is that when the frequencies are different, it is not possible to know whether the input frequency is too high or too low. A simple digital "Frequency Comparator" is shown in the diagram on the following page.

First to Point (A) in Exp. 25-1 (To set 2 KHz Reference)
Then to Point (B) in Exp. 25-1 (to measure output frequencies)



Experiment 25-3 Frequency Comparator

In the above diagram, each frequency generates a short negative or "0" pulse for each cycle. If the Test Frequency is higher than the reference frequency, the flip-flop #5 will be mostly RESET and the average output will be LOW or near "0" volts. If the Test Frequency is lower, then the Flip-Flop #5 output is mostly SET and the average output will be HIGH or near 5 volts. At the null or balance or equal frequency, the meter will swing up and down as the phase varies.

PROCEDURE (FREQUENCY COMPARATOR):

1. Retain the logic wired previously for the Counter and Phase Comparator from Experiment 25-1 and 25-2.
2. Add the logic in Experiment 25-3. Connect TEST Lead to Point (A) in 25-1. Connect the meter and the oscilloscope to the output of Flip-Flop #5; the Frequency Comparator.
3. Repeat the frequency adjustment. Set the External Generator at 2 KHz.

   NOTE: If unable to reach 2KHz, add R1 and R2 as shown in Figure 25-1.

   Vary the X POT to change the Internal Clock Frequency.

   Adjust the X POT until a null and swinging meter needle is observed.

   The Clock is now set to 2 KHz.
4. Now connect the TEST Frequency Lead (Trigger on Flip-Flop #3) to

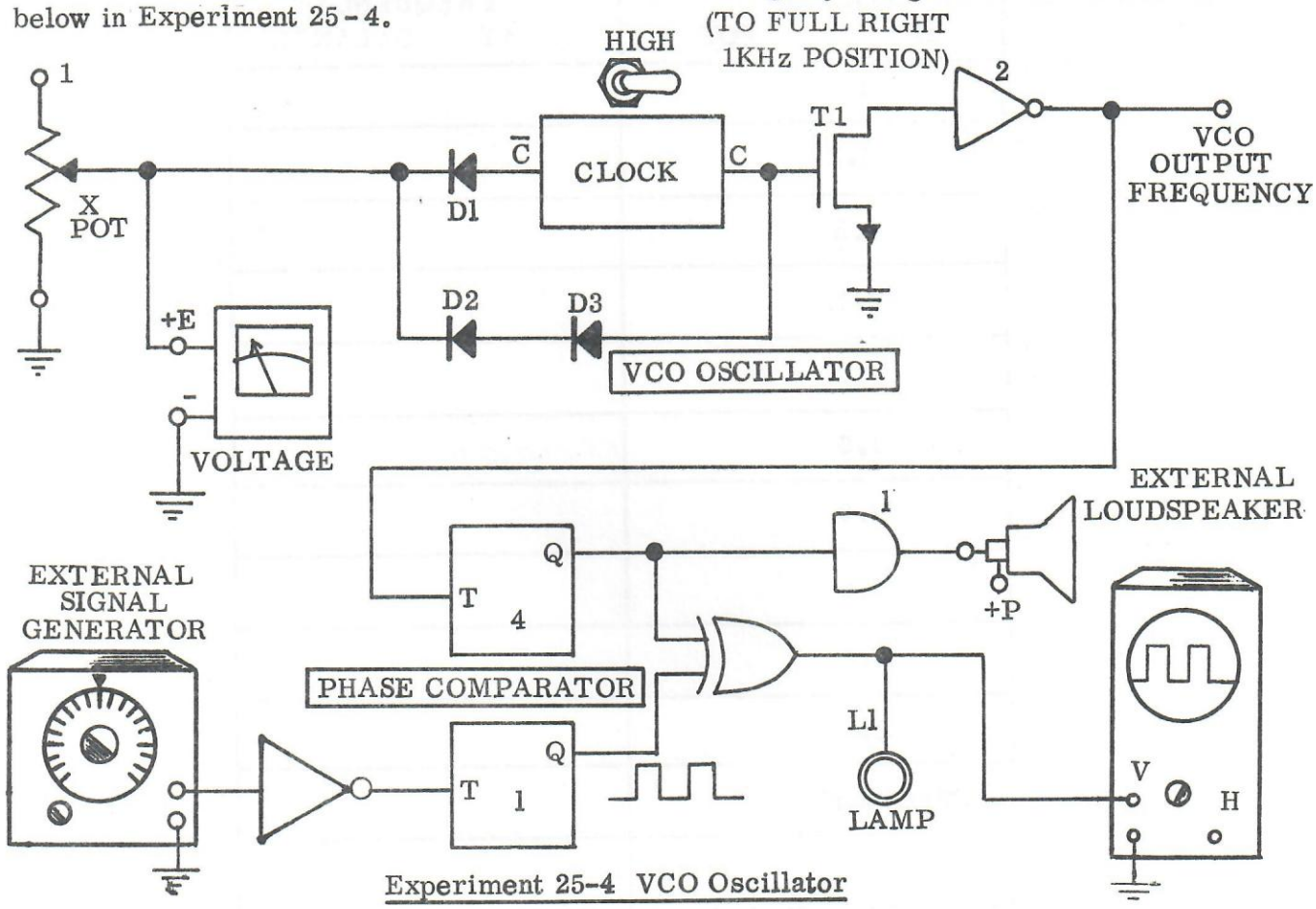
the Frequency Synthesizer Output (B) (NAND #1 OUTPUT). Set in various numbers, S, on the Counter. This determines the divider number N and the output frequency. For each number S and N, vary the external Signal Generator frequency until a null is obtained. Read the frequency on the Signal Generator and record the data below. How does this compare to the predicted values? Note that the tone heard is actually half the frequency because of Flip-Flop #7.

S	N = (16 - S)	$F_o = NF_R$	F_o (SIGNAL GENERATOR)
1111 = 15	1	2000	2000 Hz (SET)
1110 = 14	2	1000	
1101 = 13	3	667	
1100 = 12	4	500	
1011 = 11	5	400	
1010 = 10	6	333	
1001 = 9	7	286	
1000 = 8	8	250	
0111 = 7	9	222	
0110 = 6	10	200	
0101 = 5	11	182	
0100 = 4	12	167	
0011 = 3	13	154	
0010 = 2	14	143	
0001 = 1	15	133	
0000 = 0	16	125	

If the Signal Generator oscillator used has a lowest frequency of 200 Hertz, then you can add another Flip-Flop between the Signal Generator and Flip-Flop #1, and the Signal Generator frequency will be double the desired reading. Thus, for a measured frequency of 125 Hz, Signal Generator will read 250 Hz.

DESCRIPTION (VCO Oscillator):

The VCO or VOLTAGE CONTROLLED OSCILLATOR is an oscillator whose frequency is controlled by a variable voltage. The CLOCK on the Ed-Lab 700 can be frequency controlled by connecting the POTENTIOMETER to the Clock output $\overline{C}$. The Clock Frequency can also be controlled by a voltage by using two diodes as shown below in Experiment 25-4.



Experiment 25-4 VCO Oscillator

The Phase Comparator is used to detect the frequency. The Meter is used to measure the D.C. Voltage. The "blinking" light (Lamp 1) is used to indicate a frequency match. The AND element is used to drive the Loudspeaker with a low volume. The Transistor is used to amplify the small clock signal at high frequencies.

PROCEDURE (VCO):

1. Disconnect all previous experiments.

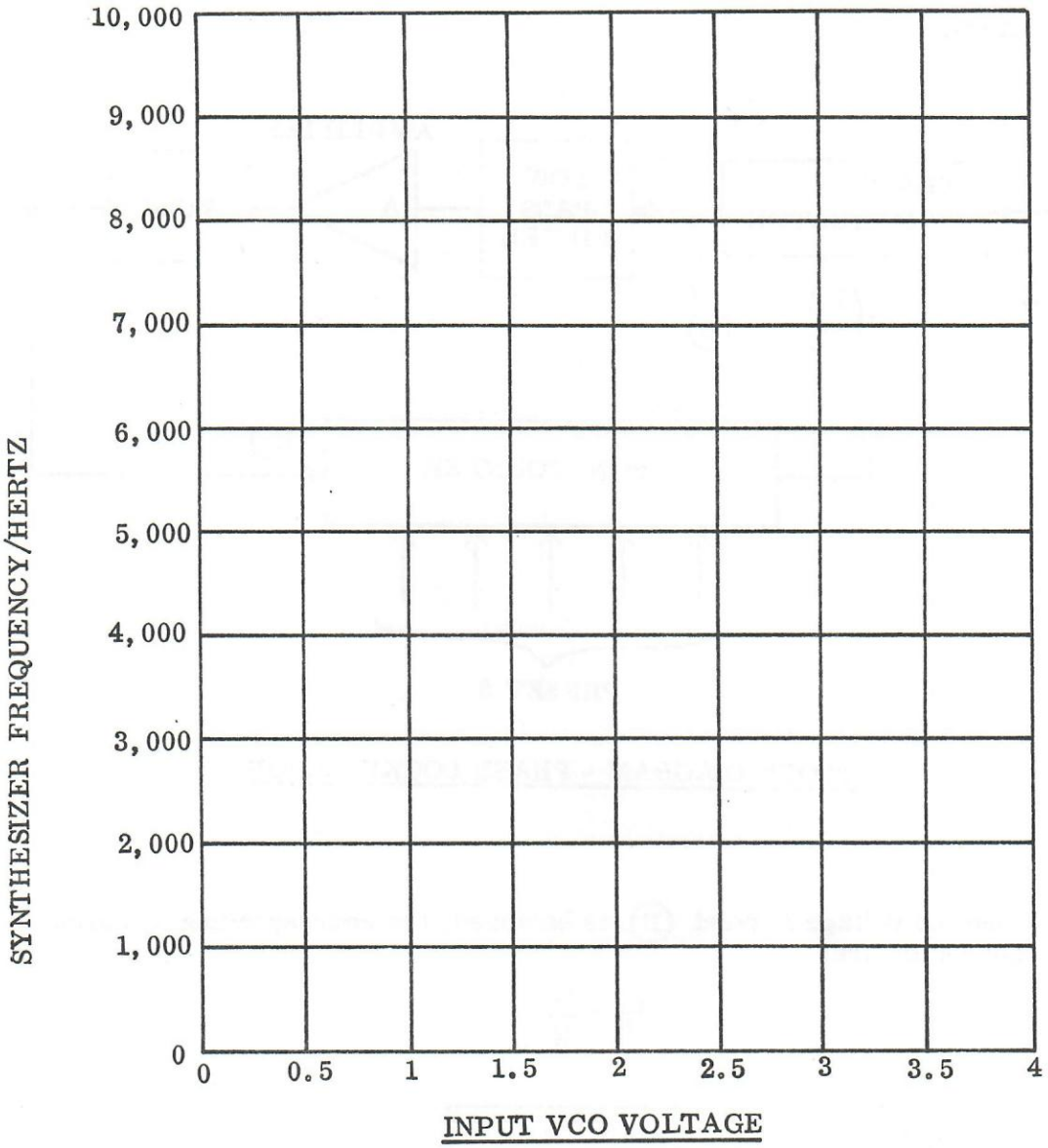
   Wire Experiment 25-4.

2. Set different voltages with the X POT. Measure the corresponding VCO frequency using the lamp. The lamp will blink near the correct frequency. This is the "beat frequency" or "difference frequency" between the matching VCO frequency and the Signal Generator frequency. When both frequencies are identical, the beat will be zero and the lamp will be steady ON or steady OFF. Therefore, the correct adjustment of the Signal Generator is where the lamp "does-not-blink" and is in-between "blinking" on slightly higher and lower frequencies.

ADJUST SIGNAL GENERATOR FREQUENCY ↓

VOLTAGE SET X POT	FREQUENCY AT BALANCE
0	
0.2	
0.5	
0.75	
1.0	
1.5	
2.0	
2.5	
3.0	
3.5	
(Maximum)	

3. Plot the results in the graph below:



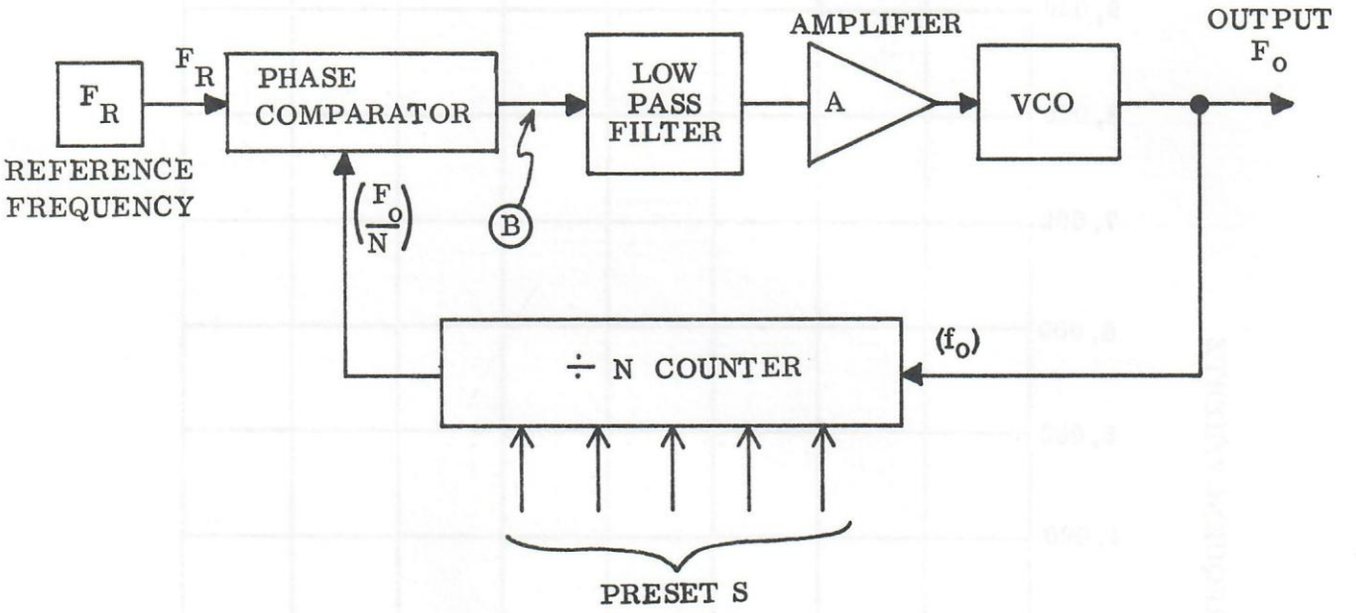
What is the range of operation?

VOLTAGE FROM _____ to _____ Volts.

FREQUENCY FROM _____ to _____ Hz.

DESCRIPTION (Phase Locked Loop)

A combination of the Preset "Divide" Counter and the "Phase-Detector" in a "closed-loop" is the Phase-Locked-Loop or PLL system which is shown below in a Block Diagram.



BLOCK DIAGRAM - PHASE LOCKED LOOP

When the voltage at point (B) is balanced, the corresponding condition is that the frequencies are:

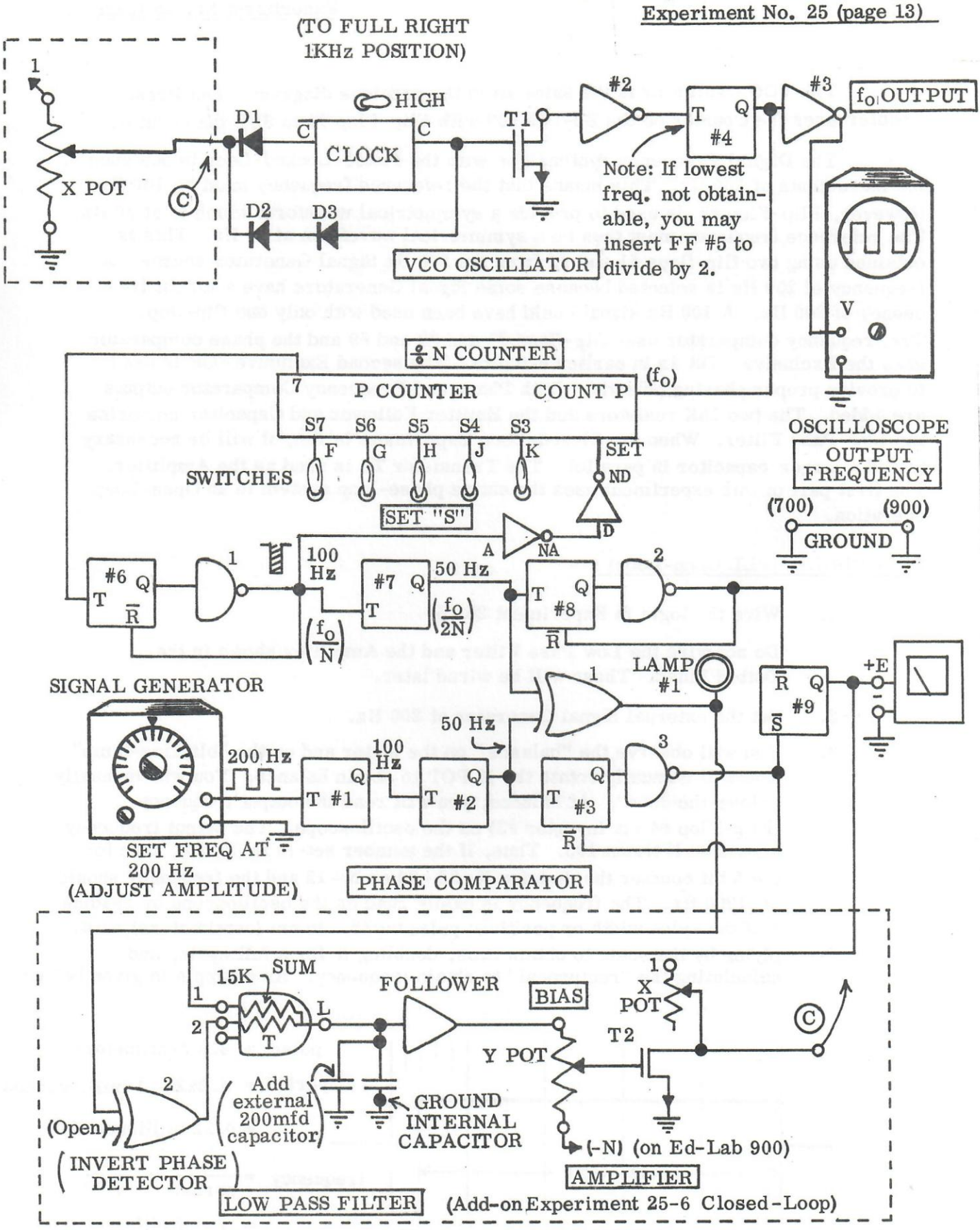
$$f_R = \frac{f_o}{N}$$

and thus:

OUTPUT FREQUENCY $\quad \boxed{f_o = Nf_R}$

The phase or polarity of feedback of the Phase-Comparator and Amplifier must be such that the system should tend to "null".

The complete logic for a Phase Locked Loop System is shown in the diagram for Experiment 25 - 5 on the following page.



Experiment 25-5 PHASE - LOCKED LOOP

The VCO Oscillator is the same as in the previous diagram. The Preset Counter uses the Counter on the Ed-Lab 900 with Flip-Flop #6 to SET the counter.

The Digital Frequency Synthesizer with the Phase-Locked-Loop is designed for increments of 100 Hz. This means that the reference frequency must be 100 Hz. However, Flip-Flop #7 is used to provide a symmetrical waveform which is at 50 Hz. The reference frequency must thus be a symmetrical waveform at 50 Hz. This is obtained using two flip-flops #1 and #2 from the 200 Hz Signal Generator source. A frequency of 200 Hz is selected because some Signal Generators have a lowest frequency of 200 Hz. A 100 Hz signal could have been used with only one flip-flop. The frequency comparator uses flip-flops #8 and #3 and #9 and the phase comparator uses the Exclusive - OR as in earlier circuits. The second Exclusive - OR is used to provide proper phasing polarity. Both Phase and Frequency Comparator outputs are added. The two 15K resistors and the Emitter Follower and Capacitor comprise the Low-Pass Filter. When the Closed-Loop experiment is run, it will be necessary to add a larger capacitor in parallel. The Transistor T2 is used as the Amplifier. The first part of this experiment uses the entire phase-loop system in an Open-Loop operation.

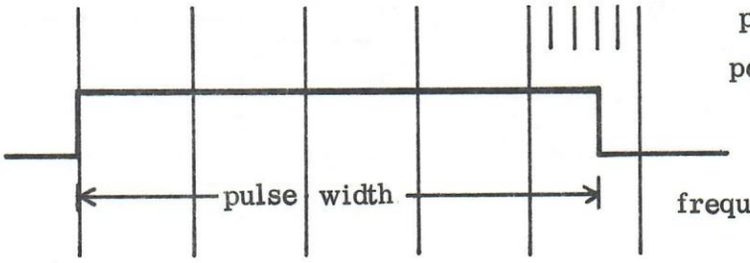
PROCEDURE (PLL Open-Loop) :

1. Wire the logic in Experiment 25 - 5.

   Do not wire the Low Pass Filter and the Amplifier shown in the Dotted Lines. These will be wired later.

2. Set the external Signal Generator at 200 Hz.

3. You will observe the "balance" on the meter and on the "blinking-lamp". You will manually rotate the X POT to obtain balance. You will manually "close-the-loop". At balance, you will read the output frequency (Flip-Flop #4 via Inverter #3) on the oscilloscope. The output frequency should be N times 100. Thus, if the number set-in is (S = 20), then for the 5 bit counter the division N = 32 - 20 or N = 12 and the frequency should be 1200 Hz. The frequency is easily read on the oscilloscope by reading half the cylce width or positive pulse width in cm (centimeters), multiplying by the scale to obtain time, doubling it for a full cycle, and calculating the "reciprocal" to obtain frequency. An example is given below:



pulse = 4.6 centimeters

period = 4.6x2x .1 milliseconds

= 0.92 milliseconds

frequency = $\frac{1}{\text{period}} = \frac{1}{0.92}$

= 1,087 KHz

4. Set-in various Switch numbers S as given in the table below.

Adjust the X POT until a balance is obtained. The Meter will swing from left to right and the lamp will blink near the correct frequency. This is the "beat-frequency" or "difference frequency" between the matching 50 Hz signal and the VCO frequency as "divided=down". When both frequencies are "matched" or equal, the beat is zero. Therefore, adjust the X POT for the frequency where the lamp "does not blink" and is in-between "blinking" operation on both sides. This results in an accurate adjustment. The table is given below for various switch settings.

Adjust the X POT for null and enter the frequencies or pulse width as measured on the oscilloscope.

Enter Data ↓ Calculate

S BINARY 16 8 4 2 1 F G H J K	S	N = (32-S)	OUTPUT F_o = N(100)Hz	Measure Time T Pulse Width	FREQ = $\frac{1}{2T \times \text{Scale}}$ Hz
0 0 0 0 0	0	32	3200		
0 0 1 0 0	4	28	2800		
0 1 0 1 0	10	22	2200		
1 0 0 0 0	16	16	1600		
1 0 1 0 0	20	12	1200		
1 0 1 1 0	22	10	1000		
1 1 0 0 0	24	8	800		
1 1 1 0 0	28	4	400		

Note that this system will only operate within the limits of the frequency variation of the VCO. The output frequencies for this system is half the frequency of the VCO because of the Flip-Flop #4 which is used to obtain a symmetrical waveform.

5. Calculate the actual output frequency from the pulse-width data measured. Enter this calculated frequency in the last column in the above table.

6. How does the PLL system frequency compare to the predicted frequency? Note that there is an error when reading frequencies on the oscilloscope corresponding to more than 0.1 cm.

   Thus, a 1000 Hz signal as read on a scale of 0.1 ms per cm has a pulse width reading of 5 cm or

$$\text{period} = 5 \text{ cm} \times 2 \times 0.1 \text{ millisec.}$$
$$= 1 \text{ millisecond}$$
$$\text{frequency} = 1000 \text{ Hz.}$$

   If an error in reading is only 0.1 cm, then the error in measurement is:

$$\text{period} = 5.1 \times 2 \times 0.1 \text{ millisec}$$
$$= 1.02 \text{ ms}$$
$$\text{frequency} = \frac{1}{\text{period}} = 980 \text{ Hz}$$

   There is a reading or measurement error of 20 Hz due to a 0.1 cm error reading on the oscilloscope. You may want to repeat some of the above measurements if there is a large error.

PROCEDURE (PLL Closed-Loop) :

Closing a loop is a very difficult task. The range of operation, the gain of the amplifier, the time delays, the biasing of the amplifiers and the polarity of the feedback must all be proper, otherwise the system will not operate. In this circuit, the Y POT and the X POT are used to vary the Amplifier Gain and Bias.

1. Remove the X POT and FOLLOWER wires from the circuit. Remove the Ground wire, the Reference Voltage (1) wire, and the Variable Voltage wire which is connected to the Follower. Remove the Follower output wire connected to the diodes.

2. Wire the Low Pass Filter, and Amplifier circuit shown in the dashed lines and labeled "Add-on Experiment 25-6". It is necessary to use an additional external capacitance as shown. You will close the loop by connecting the two points marked "C".

3. To adjust the amplifier gain and bias for the proper linear range of operation, set the number S = 1 0 1 1 0 for S = 22 and N = 10 corresponding to 1000 Hz. View the output frequency on the Oscilloscope. Set the X POT to between full (1K) and 2/3 (or 600 ohms). Now adjust the input bias Y POT until the scope display shows 1000 Hz. Typically, for a setting of 0.1 ms/cm this corresponds to 5 cm for the positive pulse. There should be a range of settings of the Y POT where this will occur where the amplifier is within the proper linear operating range.

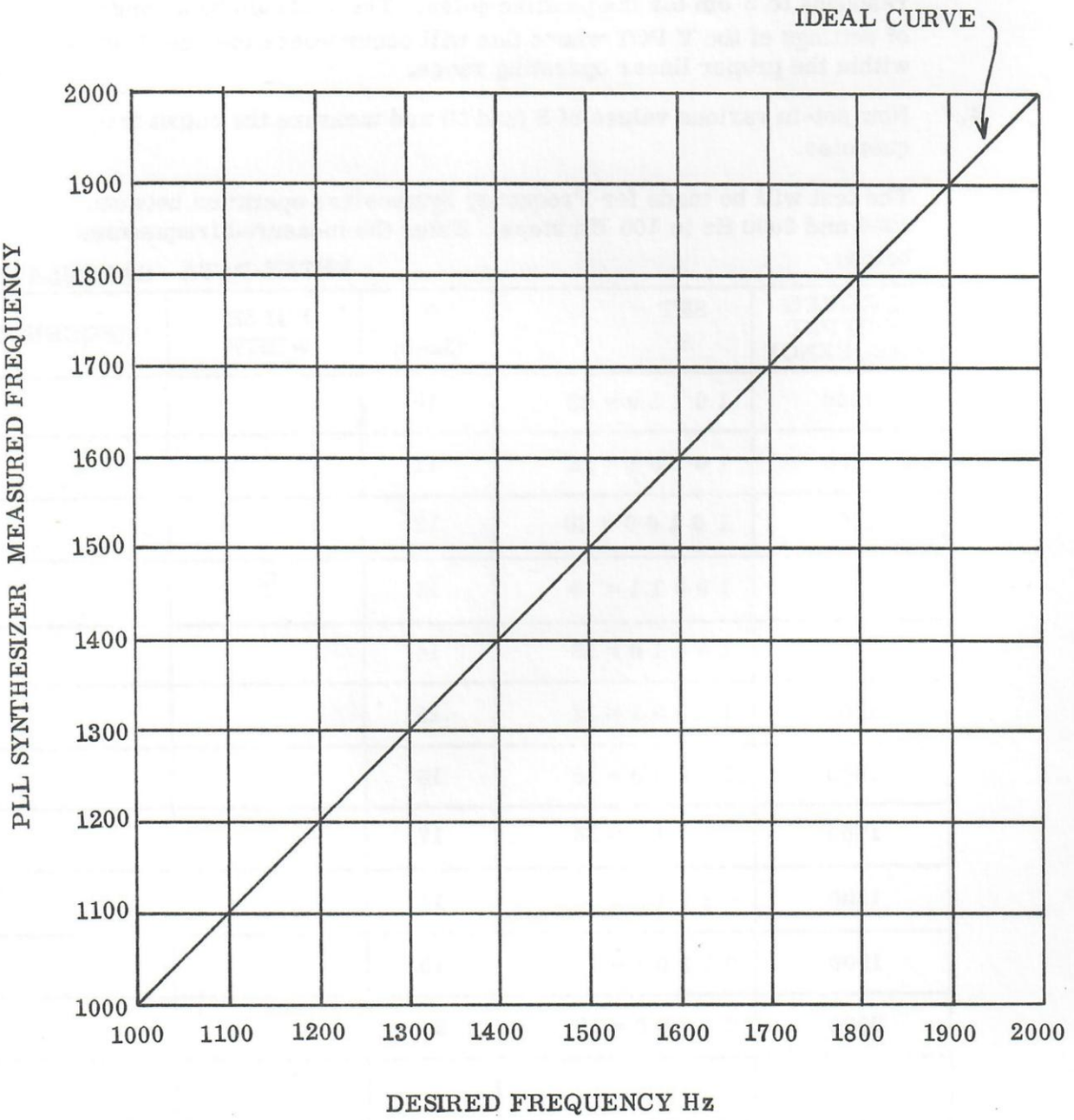
4. Now set-in various values of S (and N) and measure the output frequencies.

   The test will be made for Frequency Synthesizer operation between 1000 and 2000 Hz in 100 Hz steps. Enter the measured frequencies below:

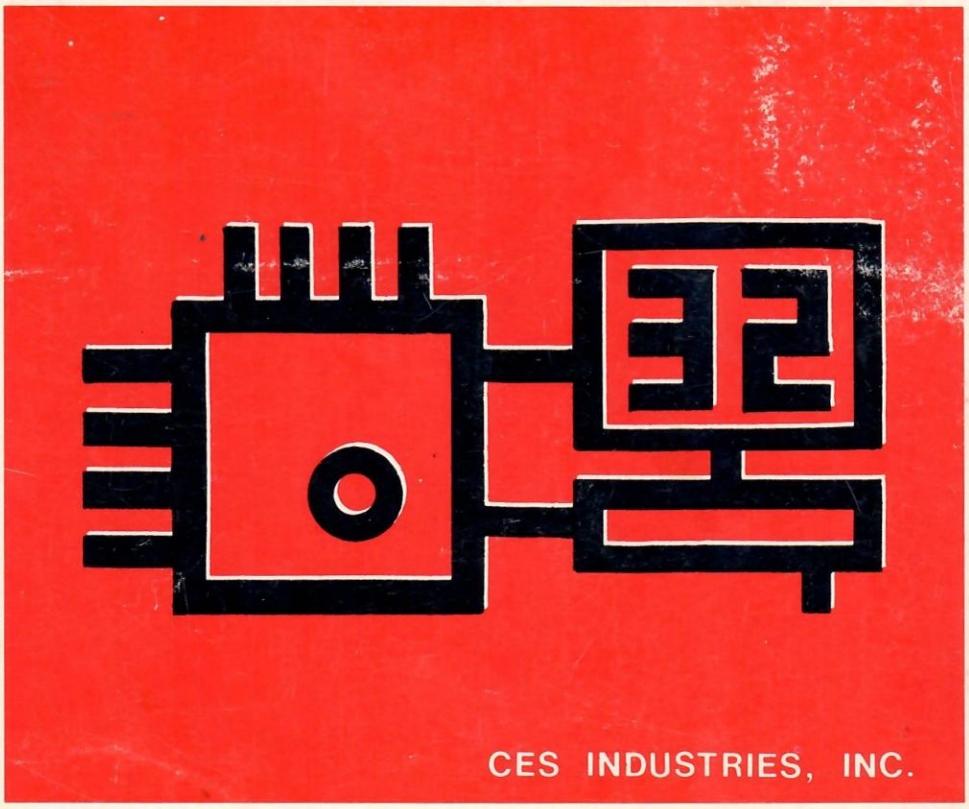
		ENTER DATA		CALCULATE
DESIRED OUTPUT FREQUENCY	SET S	N (32-S)	PULSE WIDTH	FREQUENCY
1000	1 0 1 1 0 = 22	10		
1100	1 0 1 0 1 = 21	11		
1200	1 0 1 0 0 = 20	12		
1300	1 0 0 1 1 = 19	13		
1400	1 0 0 1 0 = 18	14		
1500	1 0 0 0 1 = 17	15		
1600	1 0 0 0 0 = 16	16		
1700	0 1 1 1 1 = 15	17		
1800	0 1 1 1 0 = 14	18		
1900	0 1 1 0 1 = 13	19		
2000	0 1 1 0 0 = 12	20		

5. Calculate the measured frequencies and enter them in the last column.

6. Plot the resultant measured frequencies versus the desired frequency below:



7. What is the operating range of the Phase- Locked-Loop? Set in various other values of N to determine the overall operating range lower than 1000 Hz and greater than 2000 Hz.


CES INDUSTRIES, INC.